On Black-Box Reductions between Predicate Encryption Schemes

Vipul Goyal[*] Virendra Kumar[†] Satya Lokam[‡] Mohammad Mahmoody[§]

February 20, 2012

Abstract

We prove that there is no black-box construction of a threshold predicate encryption system from identity-based encryption. Our result signifies nontrivial progress in a line of research suggested by Boneh, Sahai and Waters (TCC '11), where they proposed a study of the relative power of predicate encryption for different functionalities. We rely on and extend the techniques of Boneh et al. (FOCS '08), where they give a black-box separation of identity-based encryption from trapdoor permutations.

In contrast to previous results where only trapdoor permutations were used, our starting point is a more powerful primitive, namely identity-based encryption, which allows planting exponentially many trapdoors in the public-key by only planting a single master public-key of an identity-based encryption system. This makes the combinatorial aspect of our black-box separation result much more challenging. Our work gives the first impossibility result on black-box constructions of any cryptographic primitive from identity-based encryption.

We also study the more general question of constructing predicate encryption for a complexity class $\mathbb{F}$, given predicate encryption for a (potentially less powerful) complexity class $\mathbb{G}$. Toward that end, we rule out certain natural black-box constructions of predicate encryption for $\mathbf{NC}^1$ from predicate encryption for $\mathbf{AC}^0$ assuming a widely believed conjecture in communication complexity.

Keywords: Predicate Encryption, Black-Box Reductions, Identity-based Encryption, Communication Complexity.

[*] Microsoft Research, India, `vipul@microsoft.com`.

[†] Georgia Tech, `virendra@cc.gatech.edu`. Work done in part while visiting Microsoft Research, India.

[‡] Microsoft Research, India, `satya@microsoft.com`.

[§] Cornell, `mohammad@cs.cornell.edu`.

1 Introduction

An encryption scheme enables a user to securely share data with other users. Traditional methods based on Secret-Key Cryptography and Public-Key Cryptography consider the scenarios where a user securely shares data with another *fixed* user whose identity (characterized by the possession of *the* decryption-key) it knows in advance. In particular, in these schemes, there is a bijection between the encryption-key and the decryption-key, fixed by the chosen encryption scheme.

As systems and networks grow in complexity, and in particular with the emergence of the cloud computing, the above viewpoint may be too narrow to cover many important applications. Often, a user might want to encrypt data to be shared with a large set of other users based on some common "property", or attribute, they satisfy. Membership in this set may not be known to the encryptor, or may not even be decidable in advance. Furthermore, a user might want to share data selectively so different users are able to decrypt different parts of that data. To cater to these scenarios, the notion of Predicate Encryption (or Attribute-based Encryption) has recently emerged. Predicate encryption was introduced by Sahai and Waters [34], and further developed in the work of Goyal et al. [19]. It has been the subject of several recent works, e.g., [1, 12, 13, 21, 26, 30, 31]. Predicate encryption is useful in a wide variety of applications; in particular, for fine-grained access control. It has also been a useful technical tool in solving seemingly unrelated problems, e.g., key escrow [18] and user revocation [7] in Identity-based Encryption (IBE). IBE [10, 14, 35] can be seen as the most basic form of a predicate encryption, where the predicate corresponds to a point function.

A predicate encryption scheme is defined in terms of a family $\mathbb{F}$ of Boolean functions (predicates) on a universe $\mathbb{A}$ of attributes. Decryption-keys are associated to a predicate $f \in \mathbb{F}$ and ciphertexts are labeled with (or are created based on) an attribute string $a \in \mathbb{A}$. A user with a decryption-key corresponding to f can decrypt a ciphertext labeled with x if and only if $f(x) = 1$. As argued by Boneh et al. [12], the key challenge in the study of predicate encryption (or Functional Encryption in general) is understanding what classes of functionalities $\mathbb{F}$ can be supported. If we could support any polynomial time computable predicate f, then any polynomial-time access control program that acts over a user's credentials could be supported [12].

Unfortunately, the current state of the art is far from being able to support an arbitrary polynomial-time f. Given this, an important direction Boneh et al. [12] suggested was to understand the relative strengths of predicate encryption schemes with respect to the functionalities they can support: When does a scheme for one functionality imply a scheme for another? In the absence of such a reduction, can we prove that predicate encryption for one functionality is inherently harder than for another? A meaningful approach to address this latter question is via *black-box separations* [20], see [29, 33] for a comprehensive survey on the topic. A proof that a cryptographic primitive P_1 cannot be constructed given black-box access to another primitive P_2 (and of course without incurring any additional assumptions) can be viewed as an indication that P_1 is in some sense a stronger primitive than P_2. Hence, to construct P_1 one may have to look for more powerful techniques, or stronger assumptions than for P_2 (or try non-black-box reductions). Thus, studying these questions would help us better understand the extent to which techniques for current predicate encryption systems might or might not be useful in obtaining systems for more general functionalities. The broad goal of this work is to make progress toward answering these questions.

Since a predicate encryption scheme has an associated family $\mathbb{F}$ of Boolean functions, a natural way to classify them is according to the complexity class this family comes from. For example, we can call a scheme $(\mathbb{A}, \mathbb{F})$ an $\mathbf{AC}^0$-PE scheme, if every member of $\mathbb{F}$ can be computed by a constant-depth polynomial size circuit (an $\mathbf{AC}^0$ circuit) on an attribute string from $\mathbb{A}$. Hence, a concrete approach to compare predicate encryption schemes is to ask questions of the kind: *Given a predicate encryption scheme for predicates in complexity class $\mathbb{G}$, can we construct a scheme for predicates in a (potentially larger) complexity class $\mathbb{F}$ in a black-box way?* For example, it is well-known that the circuit class $\mathbf{NC}^1$ is strictly larger than $\mathbf{AC}^0$. Thus a concrete question is: Is $\mathbf{NC}^1$-predicate encryption provably harder than $\mathbf{AC}^0$-predicate encryption with respect to black-box reductions? A second aspect of our work is to try to relate (perhaps conjectured) separations among Boolean function complexity classes to black-box separations among the corresponding predicate encryption schemes.

1.1 Our Results

Our main result is a black-box separation of threshold predicate encryption (TPE) from identity-based encryption (IBE) schemes. To our knowledge, this is the first result on the *impossibility* of constructing a cryptographic primitive from IBE. Recall that IBE can be viewed as the most basic form of predicate encryption in which the decryption tests exact equality (in other words, the predicate is a point function). Hence, the first natural step in the study of the above question is whether IBE can be used to construct more general predicate encryption systems. Our results show that IBE cannot be used to construct even a basic system for threshold predicates (introduced by Sahai and Waters [34]). We believe that the question of IBE vs. more advanced predicate encryption systems is of special interest. IBE as a primitive is very well studied [8–10, 14, 17, 36], and constructions of IBE are now known based on a variety of hardness assumptions.

Returning to our more general question, we rule out certain "natural" black-box constructions of predicate encryption for the class $\mathbf{NC}^1$ from predicate encryption for the class $\mathbf{AC}^0$, *assuming* a widely believed conjecture in the area of two-party communication complexity. Given black-box access to a predicate encryption scheme for $(\mathbb{B}, \mathbb{G})$, a natural way to construct a predicate encryption scheme for a "larger" system $(\mathbb{A}, \mathbb{F})$ using a a *Sharing-Based Construction* is as follows. The decryption-key for an $f \in \mathbb{F}$ is simply the set of decryption keys for a set $S(f) = \{g_1, \ldots, g_q\}$ of predicates $g_i \in \mathbb{G}$ from the smaller system. Similarly, for each attribute $a \in \mathbb{A}$, we associate a set $S(a) = \{\alpha_1, \ldots, \alpha_q\}$ of attributes from $\mathbb{B}$. To encrypt a message m under attribute a for the big system, we generate q shares $m_1, \ldots, m_q$ of m and encrypt m_j under attribute α_j of the small system. The concatenation of these encrypted shares is the ciphertext of m under a. To decrypt, we try to decrypt each m_j using the decryption keys of each $g_i \in S(f)$. The sharing construction ensures that the shares m_j that are successfully decrypted, if any, in this process suffice to recover m. Thus the sharing-based construction is a rather natural and obvious way to build predicate encryption schemes for more complex functionalities from simpler ones. Our result shows that such a sharing-based construction is impossible if $\mathbb{F}$ is a family of functions in $\mathbf{NC}^1$ and $\mathbb{G}$ is any family of functions from $\mathbf{AC}^0$, assuming certain conjectures in communication complexity. It is worth noting that combinatorial arguments about sharing-based constructions form a core component of our main result on (unrestricted) black-box separation of TPE from IBE.

1.2 Techniques

We build upon and extend the techniques of Boneh et al. [11] which rule out black-box construction of IBE from Trapdoor Permutations (TDP). Along the way, we also simplify several aspects of their proof. Given a black-box construction of TPE from IBE, our proof proceeds by designing an attack on TPE which succeeds with high probability (in fact arbitrarily close to the completeness probability of the purported TPE scheme). Somewhat more formally, we build an oracle $\mathcal{O}$ relative to which a CCA secure IBE exists, but any purported construction of a TPE relative to this oracle is insecure.

Our analysis of the attack roughly consists of a combinatorial part and a cryptographic part. The combinatorial aspect of our analysis is new and completely different from that in [11]. While the cryptographic part is similar in structure to that of [11], we do make several crucial modifications that makes our attack simpler and analysis cleaner.

A Comparison of the Combinatorial Aspects. At the heart of the proof of [11] is a combinatorial argument as follows. An IBE system obtained by a black-box construction from a TDP must embed in its public parameter the public keys of some permutations of the TDP oracle. The adversary's main goal is to collect all the trapdoors corresponding to these permutations. Such trapdoors are embedded in the decryption keys corresponding to identities in the IBE system. The main point is that there are only $q = \text{poly}(\kappa)$ many permutations planted in the public parameters of the IBE, but they must also encode an exponential number of identities. Therefore, if we look at a sufficiently large set of random identities and their secret keys, and encrypt and decrypt a random message under these identities, during at most q of these decryptions we might encounter a "new" trapdoor (which is planted in the public-key to be used during encryption, but was not discovered during other decryptions). It follows, if we choose our identity set S to be of size $k \cdot q$ (and encrypt and decrypt random messages under them), and then choose an identity $id \xleftarrow{\$} S$ at random from those $q \cdot k$ identities, then with probability at least $1 - 1/k$ there is no new (undiscovered) trapdoor left for this identity id. Therefore, whatever is learned during the decryptions of the encryptions of random messages under the identities $S \setminus \{id\}$, is sufficient to decrypt a message encrypted under id without knowing its decryption-key.

This combinatorial argument immediately suggest the following attack. Get decryption-keys for all but a random identity id_* chosen from a large enough random set $S = id_1, \dots, id_{k \cdot q}$ of identities. Collect the trapdoors learned from the encryptions of random messages under the identities in $S \setminus id_*$, and their decryptions using the corresponding decryption-keys. Try to decrypt the challenge ciphertext C encrypted under the identity id_*.

In our case, we have a related but more difficult question: what if we start with a more powerful primitive like an IBE and want to construct another "target" predicate encryption scheme? Now the intuition behind the combinatorial argument of [11] completely breaks down. The reason is that in our new setting, by planting only one (master) public-key of the IBE scheme in the public-key of the target predicate encryption, the encryption algorithm potentially has access to an *exponential* number of permutations (each indexed by an identity) whose trapdoors can be planted in the decryption-keys. In fact, each decryption-key of the predicate encryption system might have a unique trapdoor (corresponding to a unique identity derived from the description of the predicate). Hence, one can't hope to learn all trapdoors and use them to decrypt the challenge ciphertext. Thus, roughly speaking, by moving from a trapdoor permutation oracle to various forms of PE oracles such as IBE (as the primitive used in the construction), we are allowing the "universe" of trapdoor permutations planted in the public-key and decryption-keys to be exponentially large (rather than some fixed polynomial). The latter difference is the main reason behind the complications in the combinatorial aspect of our problem, because suddenly the regime of positive results becomes much richer, making the job of proving an impossibility result much more challenging.

Our proof relies on the collusion-resistance property of the predicate encryption. The "hope" that an attack exists comes from the following observations:

- The decryption key for each predicate may still consist of only a polynomial number of IBE decryption-keys.
- Each ciphertext is encrypted using a polynomially large set of identities such that a decryption-key for at least one of these identities is required to decrypt the ciphertext. On the other hand, each ciphertext can be decrypted by keys for an exponential number of different predicates (this follows from the property of a threshold encryption scheme). Call such predicates "related".
- This exponentially large set of related predicates must share an IBE decryption-key since they can decrypt a common ciphertext.
- Our attack works by requesting sufficient number of decryption-keys for related predicates (which would still be unable to decrypt challenge ciphertext). Since related predicates share IBE decryption-keys, the adversary is able to collect all "useful" IBE decryption-keys.

It is not surprising that the above combinatorial arguments sound as though they could already be used to attack sharing based constructions. Indeed, our core combinatorial lemma (Lemma 3.5) is used to refute any sharing-based construction of a TPE from an IBE (Corollary 3.10).

A Comparison of the Cryptographic Aspects. As in [11], turning the combinatorial analysis into a full-fledged impossibility result requires non-trivial black-box separation machinery. For this reason, even though the combinatorial argument of [11] is relatively simple, the full proof is quite complicated. The explanation for the complexity of such proofs is that one has to handle *all possible* constructions using a trapdoor permutation oracle (and not just where, for example, a decryption-key simply consists of decryption keys for various identities).

Although the overall structure of our proof is similar to that of [11], there are several differences in the detailed arguments. In fact, we make some crucial modifications which lead to a more direct attack and cleaner analysis.

The first major modification is that our attacker "directly" learns the heavy queries (following the paradigm of [3,4]). In [11], the attack proceeds by having steps (such as several encryptions of a random bit under the challenge identity, repeating a few steps several times) whose indirect purpose is to learn the heavy queries. Secondly, since we start with an oracle which roughly provides four functionalities (as opposed to the three functionalities of a trapdoor permutation oracle), we need to modify and adapt the techniques of [11] to to new setting Apart from these, there are significant differences in the manner we compare the various experiments which we believe makes the analysis cleaner and more general. The details regarding these can be found in Section 4.1 and beyond.

2 Preliminaries

Notation. For any probabilistic algorithm A, by $y \leftarrow A(x)$ we denote the process of executing A over the input x while using fresh randomness (which we do not represent explicitly) and getting the output y. By a *partial oracle* we refer to an oracle which is defined only for some of the queries it might be asked. By $[x \mapsto y] \in \mathcal{P}$ we mean that $\mathcal{P}(x) = y$ is defined. For a query x and a partial oracle $\mathcal{P}$, we misuse the notation and denote $x \in \mathcal{P}$ whenever an answer for x is defined in $\mathcal{P}$. By $\mathrm{Supp}(X)$ we refer to the support set of the random variable X. For every two random variables X, Y, by $\Delta(X, Y)$ we denote the statistical distance between them. For correlated random variables X and Y, by $(X \mid Y_0)$ we denote the distribution of X conditioned on $Y = Y_0$ for some $Y_0 \in \mathrm{Supp}(Y)$. For a random variable S whose values are sets, we call an element ϵ-heavy, if $\Pr[x \in S] \geq \epsilon$. The view of any probabilistic oracle algorithm A, denoted as $\mathsf{View}(A)$ refers to its input, private randomness, and oracle answers (which all together determined the whole execution of A).

2.1 Basic Probabilistic Facts

Definition 2.1. For a random variable $X = (X_1, \dots, X_k)$ consisting of k correlated random variables. For any $\ell \in [k]$ we call $y = (y_1, \dots, y_\ell)$ a prefix sample of length ℓ for X if $y_i \in \mathrm{Supp}(X_i)$ for all $i \in [\ell]$. We call an event B defined over $\mathrm{Supp}(X)$ a *prefix event*, if there is a set of prefix samples S_B for X such that $x = (x_1, \dots, x_k) \in B$ if an only if there exists some $y = (y_1, \dots, y_\ell) \in S_B$ which is a prefix of x. For any partial sequence $x^i = (x_1, \dots, x_i), i \leq k$ we say that the prefix event B happens over x^i, and denote it as the Boolean indicator $B(x^i)$, if and only if $(x_1, \dots, x_\ell) \in S_B$ for some $\ell \leq i$.

The following intuitive lemma can be verified by inspection and is implicit in many works, and here we state it formally. Roughly speaking, this lemma says: if there are two statistical games that proceed almost the same till some "bad" events happen in either of them, then their statistical distance can be bound by the probability of these bad events.

Lemma 2.2. *Let $X = (X_1, \dots, X_k)$ and $Y = (Y_1, \dots, Y_k)$ be two random variables each consisting of k correlated random variables. Let B_X and B_Y be two prefix events defined, in order, for X and Y. For every $i \in [k]$, suppose the statistical distance between $(X_{i+1} \mid (X_1, \dots, X_i) = (x_1, \dots, x_i))$ and $(Y_{i+1} \mid (Y_1, \dots, Y_i) = (y_1, \dots, y_i))$ is at most ϵ_i for every $x^i = (x_1, \dots, x_i)$ and $y^i = (y_1, \dots, y_i)$ such that: B_X does not happen over x^i and B_Y does not happen over y^i. Then it holds that $\Delta(X, Y) \leq \sum_{i \in [k]} \epsilon_i + \Pr_X[B_X] + \Pr_Y[B_Y]$.*

2.2 Predicate Encryption and Its Variants

Definition 2.3 (Predicate Encryption)**.** A predicate encryption scheme $\mathbf{PE}$ for the predicate set $\mathbb{F}_\kappa$ and attribute set $\mathbb{A}_\kappa$ with completeness ρ consists of four probabilistic polynomial time algorithms $\mathbf{PE} = (\mathbf{G}, \mathbf{K}, \mathbf{E}, \mathbf{D})$ such that for every predicate $f \in \mathbb{F}$, every attribute $a \in \mathbb{A}$ such that $f(a) = 1$, and every message M, if we do the following steps, then with probability at least ρ it holds that $M' = M$:

1. Generate a public-key and a master secret-key: $(\mathsf{PK}, \mathsf{SK}) \leftarrow \mathbf{G}(1^\kappa)$.

2. Get a decryption-key $\mathsf{DK}_f \leftarrow \mathbf{K}(\mathsf{SK}, f)$ for the predicate $f \in \mathbb{F}$.

3. Encrypt the message M under the attribute $a \in \mathbb{A}$ and get $C \leftarrow \mathbf{E}(\mathsf{PK}, a, M)$.

4. Decrypt C using the decryption-key DK_f and get $M' \leftarrow \mathbf{D}(\mathsf{PK}, \mathsf{DK}_f, C)$.

Definition 2.4 (Neighbor Sets of Predicates and Attributes)**.** For every set of predicates $\mathbb{F}$ and $f \in \mathbb{F}$, and for every set of attributes $\mathbb{A}$ and $a \in \mathbb{A}$ we define the following terminology:

- $N(f) = \{a \mid a \in \mathbb{A}, f(a) = 1\}$ and similarly $\mathbb{N}(a) = \{f \mid f \in \mathbb{F}, f(a) = 1\}$.
- $\deg(f) = |N(f)|$ and $\deg(a) = |N(a)|$.

Since we always work with families of algorithms and sets indexed by a security parameter κ, when it is clear from the context we might omit the index κ.

Definition 2.5 (Security of Predicate Encryption)**.** Let $\mathbf{PE} = (\mathbf{G}, \mathbf{K}, \mathbf{E}, \mathbf{D})$ be a predicate encryption scheme with the predicate set $\mathbb{F}$ and the attribute set $\mathbb{A}$. **PE** is said to be CPA secure if for any probabilistic polynomial time adversary **Adv** participating in the experiment below, the probability of **Adv** correctly outputting the bit b is at most $1/2 + \text{neg}(\kappa)$:

1. **Setup:** Generate the keys $(\mathsf{PK}, \mathsf{SK}) \leftarrow \mathbf{G}(1^\kappa)$ and give PK to **Adv**.
2. **Query Keys: Adv** adaptively queries some predicates $f_i \in \mathbb{F}$ for $i = 1, 2, \ldots$ and is given the corresponding decryption-keys $\mathsf{DK}_i \leftarrow \mathbf{K}(\mathsf{SK}, f_i)$.
3. **Challenge: Adv** submits an attribute $a \in \mathbb{A}$ and a pair of messages $M_0 \neq M_1$ of the same length $|M_0| = |M_1|$ conditioned on
$$f_i(a) = 0 \text{ for every predicate } f_i \text{ whose key } \mathsf{DK}_i \text{ is acquired by } \mathbf{Adv} \tag{1}$$
and is given $C \leftarrow \mathbf{E}(\mathsf{PK}, a, M_b)$ for a randomly selected $b \xleftarrow{\$} \{0, 1\}$.
4. **Adv** continues to query keys for predicates subject to condition (1) and finally outputs a bit.

PE is said to be CCA secure if for any probabilistic polynomial time adversary **Adv** participating in a modified experiment (explained next), the probability of **Adv** correctly outputting the bit b is at most $1/2 + \text{neg}(\kappa)$. The modified experiment proceeds identically as the above experiment, except that after Step 3, **Adv** is also allowed to adaptively query ciphertexts C_i for $i = 1, 2, \ldots$ encrypted under the attribute a, with the condition that $C_i \neq C$ for any i, and he is given the decrypted message $M \leftarrow \mathbf{D}(\mathsf{DK}_f, C_i)$, where $\mathsf{DK}_f \leftarrow \mathbf{K}(\mathsf{SK}, f)$ is a decryption-key for a predicate f such that $f(a) = 1$.

Comparison with [12]. The work of Boneh et al. [12] calls an attribute an *index*, calls the message M a *payload message*, and our predicate $f \in \mathbb{F}$ here is analogous to their $P(k, \cdot)$ for a *key* from the "key space" (which is the same as our predicate set $\mathbb{F}$). Also [12] considers an empty decryption-key ϵ that reveals certain information about (a, M) from $\mathbf{E}(\mathsf{PK}, a, M)$ such as the length of M (or maybe the attribute a—in which case the scheme is called *public-index*). Definition 2.5 can be deduced from the definition of [12] as a special form of functional encryption when the message length and the attribute is consciously revealed from the encryption, and thus the scheme is public index. Here we do not assume that our schemes are necessarily public index, but we will work with this weaker notion of security. This makes our negative results of Section 4 only stronger, because a stronger security definition tailored for *index hiding* schemes gives more power to the adversary as follows: **Adv** is allowed to submit $(a_0, M_0), (a_1, M_1)$ (with potentially different attributes $a_0 \neq a_1$) such that for every key DK_f acquired by him, either it holds that $f(a_0) = f(a_1) = 0$ and $|M_0| = |M_1|$, or that $f(a_0) = f(a_1) = 1$ and $M_0 = M_1$.

Definition 2.6 (Identity-based Encryption [35])**.** An Identity Based Encryption scheme is a predicate encryption scheme where **(1)** the predicate and attribute sets are equal $\mathbb{A} = \mathbb{F} = \{0,1\}^\kappa$ (and are called the set of identities), and **(2)** for every predicate $f \in \{0,1\}^\kappa$ and every attribute $a \in \{0,1\}^\kappa$ we have that $f(a) = 1$ if and only if $f = a$.

Definition 2.7 (Threshold Predicate Encryption [34])**.** A Threshold Predicate Encryption with threshold $0 < \tau < 1$ (or simply a τ-**TPE**) is a predicate encryption where both the predicate and the attribute sets are equal to $\{0,1\}^\kappa$ and for any predicate $f \in \{0,1\}^\kappa$ and any attribute $a \in \{0,1\}^\kappa$ we have that $f(a) = 1$ if and only if $\langle f, a\rangle \geq \tau \cdot \kappa$ where $\langle f, a\rangle$ is the inner product of the Boolean vectors $f = (f_1, \ldots, f_\kappa), a = (a_1, \ldots, a_\kappa)$ defined as $\langle f, a\rangle = \sum_{i\in[\kappa]} a_i \cdot f_i$.

The notion of threshold predicate encryption was defined by [34] and is also known as the *fuzzy* IBE.

3 Sharing-based Constructions and Impossibility Results

In this section, we describe two intuitive and simple approaches to build a predicated encryption scheme using another predicate encryption scheme as a black-box. It is interesting that the simpler of the two, the OR-based approach turns out to be as powerful as the seemingly more general Sharing-based approach. Even though ruling out constructions using these approaches is a weaker impossibility result than an unrestricted black-box separation (as we will do in Section 4), it seems instructive to refute these natural and general approaches to black-box reductions among predicate encryption schemes. In fact, our proof refuting OR-based constructions of TPE in Section 3.3 forms the combinatorial core of our subsequent proof of a general black-box separation in Section 4. Moreover, the basic approach to building the attack needed in our proof (as well as that in [11]) of the general black-box separation results seems to benefit by keeping the sharing-based constructions in mind. In Section 3.4, we investigate a new approach to refute sharing-based constructions using (proved or conjectured) separation results in two-party communication complexity. In particular, we can use conjectures in communication complexity to give evidence that $\mathbf{NC}^1$-predicate encryption is strictly harder than $\mathbf{AC}^0$-predicate encryption.

3.1 The OR-Based Approach

Definition 3.1. Let $(\mathbb{F}, \mathbb{A})$ and $(\mathbb{G}, \mathbb{B})$ be two pairs of predicate and attribute sets. We call $S(\cdot)$ a *q-set system* for $(\mathbb{F}, \mathbb{A})$ using $(\mathbb{G}, \mathbb{B})$ if S is a mapping defined over $\mathbb{F} \cup \mathbb{A}$ such that:

1. For every $f \in \mathbb{F}$ it holds that $S(f) \subset \mathbb{G}$, and for every $a \in \mathbb{A}$ it holds that $S(a) \subset \mathbb{B}$.

2. For every $x \in \mathbb{F} \cup \mathbb{A}$ it holds that $|S(x)| \leq q$.

Definition 3.2. We say there is an OR-based construction with set-size q for the pair of predicate and attribute sets $(\mathbb{F} = \{f_1, \ldots\}, \mathbb{A} = \{a_1, \ldots\})$ using another pair $(\mathbb{G} = \{\varphi_1, \ldots\}, \mathbb{B} = \{\alpha_1, \ldots\})$ if there exists a q-set system $S(\cdot)$ for $(\mathbb{F}, \mathbb{A})$ using $(\mathbb{G}, \mathbb{B})$ such that the following holds:

- For every $f \in \mathbb{F}$ and $a \in \mathbb{A}$, if $S(f) = \{\varphi_1, \ldots, \varphi_{d_f}\}$ and $S(a) = \{\alpha_1, \ldots, \alpha_{d_a}\}$, then $f(a) = \bigvee_{i\in[d_f], j\in[d_a]} \varphi_i(\alpha_j)$.

We call the OR-based construction *efficient* if the mapping $S(\cdot)$ is efficiently computable.

Lemma 3.3. *Suppose there exists an efficient* OR*-based construction for* $(\mathbb{F}, \mathbb{A})$ *using* $(\mathbb{G}, \mathbb{B})$*. Then a secure predicate encryption scheme* $\mathbf{PE}_1 = (\mathbf{G}_1, \mathbf{K}_1, \mathbf{E}_1, \mathbf{D}_1)$ *for* $(\mathbb{F}, \mathbb{A})$ *with completeness* ρ *can be constructed (in a black-box way) from any secure predicate encryption scheme* $\mathbf{PE}_2 = (\mathbf{G}_2, \mathbf{K}_2, \mathbf{E}_2, \mathbf{D}_2)$ *for* $(\mathbb{G}, \mathbb{B})$ *with completeness* ρ*.*

Proof. First we describe how the components of $\mathbf{PE}_1$ work and then will prove its security.

- $\mathbf{G}_1(1^\kappa)$**:** The public-key and master secret-key are sampled as $(\mathsf{PK}, \mathsf{SK}) \leftarrow \mathbf{G}_2(1^{\kappa_2})$ (for the right value of the security parameter κ_2, chosen based on the main security parameter κ).
- $\mathbf{K}_1(\mathsf{SK}, f)$**:** To get a decryption-key DK_f for any predicate $f \in \mathbb{F}$, first compute $S(f) = \{\varphi_1, \dots, \varphi_{d_f}\} \subseteq \mathbb{G}$, then get the keys $\mathsf{DK}_i \leftarrow \mathbf{K}_2(\mathsf{SK}, \varphi_i)$ for every $i \in [d_f]$, and finally take $\mathsf{DK}_f = (f, \mathsf{DK}_1, \dots, \mathsf{DK}_{d_f})$.
- $\mathbf{E}_1(\mathsf{PK}, a, m)$**:** To get the encryption $c \leftarrow \mathbf{E}_1(\mathsf{PK}, a, M)$ for any attribute $a \in \mathbb{A}$ and message M, first compute $S(a) = \{\alpha_1, \dots, \alpha_{d_a}\} \subseteq \mathbb{B}$, then encrypt M independently under α_i for every $i \in [d_a]$ to get $C_i \leftarrow \mathbf{E}_2(\mathsf{PK}, \alpha_i, M)$, and finally take $C = (a, C_1, \dots, C_{d_a})$.
- $\mathbf{D}_1(\mathsf{PK}, \mathsf{DK}_f, C)$**:** To decrypt the ciphertext $C = (a, C_1, \dots, C_{d_a})$ under the key $\mathsf{DK}_f = (f, \mathsf{DK}_1, \dots, \mathsf{DK}_{d_f})$, first get the sets $S(a) = \{\alpha_1, \dots, \alpha_{d_a}\}, S(f) = \{\varphi_1, \dots, \varphi_{d_f}\}$, and look for the lexicographically first pair (i, j) such that $\varphi_i(\alpha_j) = 1$ and output $\mathbf{D}_2(\mathsf{PK}, \mathsf{DK}_i, C_j)$.

The ρ-completeness of $\mathbf{PE}_1$ is simply inherited from $\mathbf{PE}_2$. In what follows we will prove the security of $\mathbf{PE}_1$. Given an adversary $\mathbf{Adv}_1$ attacking $\mathbf{PE}_1$ and succeeding with probability $1/2 + \mu$ (we call μ the advantage of the adversary), we construct another adversary $\mathbf{Adv}_2$ attacking $\mathbf{PE}_2$ with advantage at least μ/q where $q = \mathrm{poly}(\kappa)$ is the set-size of the set-system $S(\cdot)$. Moreover, if $\mathbf{Adv}_1$ runs in polynomial time, then so does $\mathbf{Adv}_2$ who makes at most q times the number of queries made by $\mathbf{Adv}_1$. First we consider a security game in which an adversary $\mathbf{Adv}'$ can submit up to $2k \leq 2q$ (rather than two) pairs:

$$(\alpha_1, M_0^1), (\alpha_1, M_1^1), \dots, (\alpha_k, M_0^k), (\alpha_k, M_1^k)$$

such that every $(\alpha_i, M_0^i), (\alpha_i, M_1^i)$ satisfy the condition (1). In return $\mathbf{Adv}'$ will get either the encryptions of (α_i, M_0^i) for all $i \in [k]$ or the encryptions of (α_i, M_1^i) for all i, and is supposed to guess which encryptions he gets. We call the latter an extended security game. It can be seen through a standard hybrid argument (*cf.*, Theorem 11.3.1, Chapter 11 of [5]) that if $\mathbf{Adv}'$ can have advantage μ' in this game, then another adversary $\mathbf{Adv}$ can get advantage at least μ'/q according to Definition 2.5.

Now, we will construct an adversary $\mathbf{Adv}_2'$ attacking $\mathbf{PE}_2$ in the extended security game with advantage μ. By the discussion above it can be turned into a standard adversary $\mathbf{Adv}_2$ of advantage μ/q according to Definition 2.5. $\mathbf{Adv}_2'$ acts as follows.

- **Setup:** $\mathbf{Adv}_2'$ gets PK and initiates the simulation of $\mathbf{Adv}_1$ by giving PK to it.
- **Query:** For any query $f \in \mathbb{F}$ that $\mathbf{Adv}_1$ makes, $\mathbf{Adv}_2'$ first computes $S(f) = \{\varphi_1, \dots, \varphi_{d_f}\}$, then queries all of $\varphi_1, \dots, \varphi_{d_f}$ and gets back keys $\mathsf{DK}_1, \dots, \mathsf{DK}_{d_f}$, and finally returns $\mathsf{DK}_f = (f, \mathsf{DK}_1, \dots, \mathsf{DK}_{d_f})$ to $\mathbf{Adv}_1$.

- **Challenge:** When $\mathbf{Adv}_1$ submits two pairs $(a, M_0), (a, M_1)$ (satisfying the condition $f(a) = 0$ for every query f made by $\mathbf{Adv}_1$), then $\mathbf{Adv}'_2$ computes $S(a) = \{\alpha_1, \dots, \alpha_{d_a}\}$. Then for all $i \in [d_a]$ it submits $(\alpha_i, M_0), (\alpha_i, M_1)$, gets C_i, and returns $C = (a, C_1, \dots, C_{d_a})$ to $\mathbf{Adv}_1$.

- If $\mathbf{Adv}_1$ makes more queries, then $\mathbf{Adv}'_2$ continues answering them as before and finally, $\mathbf{Adv}'_2$ outputs whatever bit $\mathbf{Adv}_1$ outputs.

Note that the condition (1) in the challenge phase is satisfied for $\mathbf{Adv}'_2$ if it is satisfied for $\mathbf{Adv}_1$. From Definition 3.2, if for any $f \in \mathbb{F}$ and $a \in \mathbb{A}$ $f(a) = 0$, then for all $i \in [d_f]$ and $j \in [d_a]$ $\varphi_i(\alpha_j) = 0$, where $S(f) = \{\varphi_1, \dots, \varphi_{d_f}\}$ and $S(a) = \{\alpha_1, \dots, \alpha_{d_a}\}$. Thus, if $f(a) = 0$ for every query f made by $\mathbf{Adv}_1$, then so is true for $\mathbf{Adv}'_2$, *i.e.*, $\varphi(\alpha_1) = \cdots = \varphi(\alpha_{d_a}) = 0$ for every query φ made by $\mathbf{Adv}'_2$. Now, it is easy to see that $\mathbf{Adv}'_2$ is correctly simulating the experiment for $\mathbf{Adv}_1$ and therefore achieves the same advantage μ. □

3.2 The Sharing-Based Approach

Clearly, the OR-based construction of Lemma 3 is not the only way that one can imagine to construct a $\mathbb{F}$-**PE** from a $\mathbb{G}$-**PE**. In fact, as noted also by [22] in the context of using trapdoor permutations, there is a possibility of employing a more complicated "sharing-based" approach that generalizes the OR-based construction of Lemma 3.3. The idea is to use a set system $S(\cdot)$ in a similar way to the OR-based construction, but to encrypt the message M differently: Instead of encrypting the message M d_a times, first construct some "shares" $M_1, \dots, M_{d_a}$ of M, and then encrypt each M_i using α_i. To get the completeness and the security, we need the following two properties.

- Completeness: For every $f \in \mathbb{F}$ such that $f(a) = 1$, the set of indices $I_S(a, f) = \{j \mid \exists\varphi \in S(f)$ such that $\varphi(\alpha_j) = 1\}$ is rich enough so that $\{M_i \mid i \in I_S(a, f)\}$ can be used to reconstruct M.

- Security: For every choice of $a_*, f_*, f_1, \dots, f_k$ for $k = \mathrm{poly}(\kappa)$ such that $f_*(a_*) = 1$ and $f_i(a_*) = 0$ for all $i \in [k]$, it holds that $C_S(a_*, f_*) \not\subseteq \cup_{j\in[k]} C_S(a_*, f_j)$, where $C_S(a, f) = \{\alpha_i \mid i \in I_S(a, f)\}$. This is because otherwise the adversary can acquire keys for $f_1, \dots, f_k$ and use the sub-keys planted in them to decrypt enough number of shares of M_i's and reconstruct M which is encrypted under the attribute a_*.

As we will see, despite the fact that the sharing-based approach is more general than the OR-based approach, for the case of polynomial sized sets $q = \mathrm{poly}(\kappa)$, the construction of Lemma 3.3 is indeed as powerful as any sharing-based approach.

Lemma 3.4. *There is a sharing based construction for the predicate system $\mathbb{F}$ using $\mathbb{G}$ if and only if there exists an OR-based construction.*

Proof of Lemma 3.4. First suppose there is an OR-based construction for $\mathbb{F}$ using $\mathbb{G}$ using the set system $S(\cdot)$. By Definition 3.2 and the definition of the sharing based construction $S(\cdot)$ is already a sharing based construction as well, because $C_S(a_*, f_*) \neq \varnothing$, but $C_S(a_*, f_j) = \varnothing$ for all $j \in [k]$ which implies the condition required for the sharing-based construction.

Now suppose that $S(\cdot)$ is a sharing based construction with set sizes $q = \mathrm{poly}(\kappa)$ for $\mathbb{F}$ using $\mathbb{G}$. Define the set system $T(\cdot)$ as follows: For every pair $a \in \mathbb{A}, f \in \mathbb{F}$ such that $f(a) = 0$ and recall the set $C_S(a, f) = \{\alpha \mid \alpha \in S(a), \exists\varphi \in S(f), \varphi(\alpha) = 1\}$, and update a new value for $S(a)$ according to $S(a) \leftarrow S(a) \setminus C_S(a, f)$. After all these changes (which do not apply to any $S(f)$) take $T(\cdot)$ to be the new updated set system $S(\cdot)$. Now we claim that $\mathbb{G}$ can compute $\mathbb{F}$ using the set system $T(\cdot)$:

- First look at any pair $a \in \mathbb{A}, f \in \mathbb{F}$ such that $f(a) = 0$. Then by the definition of $T(\cdot)$ we have already removed the set $C_S(a, f)$ from $T(\cdot)$ and therefore it holds that

$$\bigvee_{\alpha \in S(a), \varphi \in S(f)} \varphi(\alpha) = 0.$$

- Now consider any pair $a \in \mathbb{A}, f \in \mathbb{F}$ such that $f(a) = 1$ and suppose for sake of contradiction that $\bigvee_{\alpha \in T(a), \varphi \in T(f)} \varphi(\alpha) = 0$, which is equivalent to $C_T(a, f) = \varnothing$. Since $S(\cdot)$ was of set-size q, then we can always find a sequence $f_1, \ldots, f_k$ of length $k \leq q$ (or even potentially $k = 0$) such that they can "represent" the changes in $S(a)$, namely $T(a) = S(a) \setminus \cup_{i \in [k]} C_S(a, f)$, which together with $C_T(a, f) = \varnothing$ and $T(f) = S(f)$ implies that

$$C_S(a, f) \subset \bigcup_{i \in [k]} C_S(a, f_i).$$

Note that also by definition of the way we changed $S(\cdot)$, for all $i \in [k]$ holds that $f_i(a) = 0$. Thus, the set $\{a, f, f_1, \ldots, f_k\}$ violates the security requirement of the sharing-based method.

□

3.3 A Combinatorial Argument Refuting OR-Based Constructions

Recall that by proving Theorem 4.1 we shall rule out the OR-based (and sharing-based) constructions of Section 3 along the way. A special case of the following combinatorial lemma shows that no OR-based (nor sharing-based) construction of τ-**TPE** from **IBE** exists for any constant $0 < \tau < 1$. Moreover, not surprisingly, we will use this lemma in our proof of Theorem 4.1.

Lemma 3.5. *Let $\mathbb{F} = \mathbb{A} = \{0,1\}^\kappa$ denote the set of attributes and predicates for τ-**TPE** for a constant $0 < \tau < 1$. Also suppose that the following sets of size at most $q = \operatorname{poly}(\kappa)$ are assigned to $\mathbb{F}$, $\mathbb{A}$, and $\mathbb{F} \times \mathbb{A}$: $S(a)$ for $a \in \mathbb{A}$, $S(f)$ for $f \in \mathbb{F}$, and $S(a, f)$ for $(a, f) \in \mathbb{A} \times \mathbb{F}$.* [1] *Then, there exists a sampling algorithm* Samp *that, given an input parameter $\epsilon > 1/\operatorname{poly}(\kappa)$, outputs $k + 1 = \operatorname{poly}(\kappa)$ pairs $(f_*, a_*), (f_1, a_1), \ldots, (f_k, a_k)$ such that with probability at least $1 - \epsilon$ over the randomness of* Samp *the following holds:*

1. *$f_*(a_*) = 1$ and $f_i(a_i) = 1$ for all $i \in [k]$ (this part holds with probability 1),*
2. *$f_i(a_*) = 0$ for all $i \in [k]$,*
3. *$S(a_*) \cap S(f_*) \cap S(a_*, f_*) \subseteq \bigcup_{i \in [k]} S(a_i, f_i)$.*

Moreover, the algorithm Samp *chooses its $k + 1$ pairs* without *the knowledge of the set system $S(\cdot)$. Therefore we call* Samp *an* oblivious *sampler against predicate structure of τ-**TPE**.*

[1] Note that although $\mathbb{F} = \mathbb{A}$, the sets $S(a)$ for $a \in \mathbb{A}$ and $S(f)$ for $f \in \mathbb{F}$ are potentially different even if a and f represent the same string. Intuitively the set $S(a)$ refers to the set of sub-attributes (or identities in case of using IBE as the black-box primitive) used during an encryption of a random message under the attribute a, the set $S(f)$ refers to the set of decryption-keys planted in the decryption-key of f, and finally $S(a, f)$ refers to the decryption-keys discovered during the decryption of the mentioned random encryption (under the attribute a) using the generated key for f.

Proof of Lemma 3.5. Let $\mathcal{A}$ be the set of vectors in $\{0,1\}^\kappa$ of normalized Hamming weight τ, namely $\mathcal{A} = \{a \mid a = (a_1, \ldots, a_\kappa) \in \{0,1\}^\kappa, \sum_i a_i = \tau \cdot \kappa\}$. Also let $\mathcal{F}$ be the set of vectors in $\{0,1\}^\kappa$ of normalized Hamming weight $\tau' = \tau + \frac{1-\tau}{2}$. Consider a bipartite graph G with nodes $(\mathcal{A}, \mathcal{F})$ and connect $a \in \mathcal{A}$ to $f \in \mathcal{F}$ iff $f(a) = 1$ according to τ-**TPE** (i.e., the indexes of the nonzero components of a is a subset of those of f). We will later use the fact that G is a regular graph (on its $\mathcal{F}$ side). For any vertex x in G let $N(x)$ be the set of neighbors of x in the graph G.

The covering-sampler acts as follows:

- Choose $p = \mathrm{poly}(\kappa)$ and $h = \mathrm{poly}(\kappa)$ to satisfy $q(\frac{1}{p} + \frac{1}{h} + (1 - \frac{1}{h})^p) < \frac{\epsilon}{2}$ (e.g., this can be done by setting $h = \sqrt{p}$ and choosing p large enough).
- Choose $f_* \xleftarrow{\$} \mathcal{F}$ at random.
- Choose $a_*, a_1, \ldots, a_p \xleftarrow{\$} N(f_*)$ at random *with possible repetition* from the neighbors of f_*.
- For each $i \in [p]$, choose p random neighbors $f_{i1}, \ldots, f_{ip} \xleftarrow{\$} N(a_i)$ of a_i (repetition is allowed).
- Output the $p^2 + 1$ pairs: $(a_*, f_*), (a_i, f_{ij})_{i \in [p], j \in [p]}$.

Now we prove that with probability at least $1 - \epsilon/2 - \mathrm{neg}(\kappa) > 1 - \epsilon$ the output pairs have the properties specified in Lemma 3.5.

Property (1) holds by construction.

Since $0 < \tau < \tau' < 1$ are constants, using standard probabilistic arguments one can easily show that the probability of f_{ij} being connected to a_* in G (i.e., $f_{ij}(a_*) = 1$) is $\mathrm{neg}(\kappa)$ (given a_*, a_i are random subsets of f_*, a random superset f_{ij} of a_i is exponentially unlikely to pick all the elements of a_*). Thus (2) holds.

The challenging part is to show that (3) holds, i.e., the following: With probability at least $1 - q(\frac{1}{p} + \frac{1}{\sqrt{p}} + (1 - \frac{1}{\sqrt{p}})^p) \geq 1 - \epsilon/2$ it holds that $S(a_*) \cap S(f_*) \cap S(a_*, f_*) \subset \cup_{ij} S(a_i, f_{ij})$. The proof will go through several claims.

In the following let $h = \sqrt{p}$.

For an attribute node $a \in \mathcal{A}$ of G, define $H(a)$ to be the set of "heavy" elements that with probability at least $1/h$ are present in $S(a, f)$ for a random neighbor f of a, i.e.,

$$H(a) = \{x \colon \Pr[x \in S(a, f) \mid f \xleftarrow{\$} N(a)] > 1/h\}.$$

Note that $H(a)$ is not necessarily a subset of $S(a)$.

Claim 3.6. *Define* BE_1 *to be the bad event "$S(a_*) \cap S(a_*, f_*) \not\subseteq H(a_*)$." Then,* $\Pr[\mathsf{BE}_1] \leq q/h$.

Proof. Since G is regular on its $\mathbb{F}$ side, conditioned on a fixed a_* the distribution of f_* is still uniform over $N(a_*)$. Now fix a_* and fix an element $b \in S(a_*)$. If b is not in $H(a_*)$, then over the random choice of $f_* \xleftarrow{\$} N(a_*)$, it holds that $\Pr[b \in S(a_*, f_*)] \leq 1/h$. The claim follows by a union bound over the q elements in $S(a_*)$. □

Claim 3.7. *Define* BE_2 *to be the bad event "there exists a $b \in S(f_*)$ such that $b \in H(a_*)$ but for every $i \in [p]$, $b \notin H(a_i)$, i.e., $S(f_*) \cap H(a_*) \not\subseteq \cup_i H(a_i)$." Then,* $\Pr[\mathsf{BE}_2] \leq q/p$.

Proof. It is enough to bound BE_2 by $1/p$ for a fixed $b \in S(f_*)$ and the claim follows by union bound over the elements of $S(f_*)$. But when $b \in S(f_*)$ is fixed, we can pretend that a_* is chosen at random from the sequence $a_0, \dots, a_p$ *after* they are chosen and are fixed. In that case BE_2 happens if there is only a unique $j \in \{0, \dots, p\}$ such that $b \in H(a_j)$ and a_* chooses to be a_j. The latter happens with probability at most $1/(p+1) < 1/p$. □

Claim 3.8. *Define* BE_3 *to be the bad event "*given neither BE_1 nor BE_2 happens*,* $S(a_*) \cap S(f_*) \cap S(a_*, f_*) \not\subseteq \cup_{i,j} S(a_i, f_{ij})$*." Then,* $\Pr[\mathsf{BE}_3] \leq q(1 - 1/h)^p$.

Proof. We assume that the events BE_1 and BE_2 have not happened and do the analysis. By $\neg\mathsf{BE}_1$ we have $S(a_*) \cap S(a_*, f_*) \subseteq H(a_*)$. Moreover, since $\neg\mathsf{BE}_2$ holds, any element $b \in S(f_*) \cap H(a_*)$ will be in $H(a_i)$ for at least one value $i \in [p]$. Therefore for each $j \in [p]$, $\Pr[b \in S(a_i, f_{ij})] \geq 1/h$ holds by the definition of heavy sets, and thus $b \notin \cup_j S(a_i, f_{ij})$ can hold only with probability at most $(1 - 1/h)^p$. By union bound, the probability that there exists a $b \in S(a_*) \cap S(f_*) \cap S(a_*, f_*)$ such that $b \notin \cup_j S(a_i, f_{ij})$ is bounded by $q(1 - 1/h)^p$. □

From Claims 3.6, 3.7, and 3.8, it follows that (3) fails with probability at most $q(\frac{1}{p} + \frac{1}{h} + (1 - \frac{1}{h})^p) < \frac{\epsilon}{2}$.

Therefore the sampled $[a_*, f_*, \{f_{ij}\}_{i \in [p], j \in [p]}]$ will have the desired properties with probability at least $1 - \mathrm{neg}(\kappa) - \epsilon/2$ which finishes the proof of Lemma 3.5. □

Remark 3.9 (Generalizing Lemma 3.5)**.** We observe that, by going over the proof of Lemma 3.5 more carefully, it can be realized that the only required properties for the pair of sets $(\mathbb{F}, \mathbb{A})$ to make the lemma hold are the following:

1. $\deg(f) = \deg(f')$ for every $f, f' \in \mathbb{F}$, and
2. for a random $f \in \mathbb{F}$ and two of its random neighbors $a, a' \in N(f)$, the probability that a and a' have another common neighbor $f' \in \mathbb{F}$ is a negligible function of κ. Roughly speaking this says that the occurrence of a $K_{2,2}$ in the bipartite graph G is unlikely or a random path of length 3 is unlikely to be completed into a C_4.

Even though the proof of the black-box impossibility of the next section would refute any positive approach that includes as a special case the OR-based and sharing-based constructions, for sake of clarity, we first point out directly why Lemma 3.5 refutes those constructions.

Corollary 3.10. *For any constant* $0 < \tau < 1$*, there is no* OR*-based (nor sharing-based) construction of* τ*-TPE schemes from IBE schemes. The claim holds also for any predicate encryption scheme (other than* τ*-TPE) with the properties specified in Remark 3.9.*

Proof. We directly refute sharing-based constructions and OR-based constructions are refuted as well since every OR-based construction can be thought of a sharing-based construction as well.

Suppose $S(\cdot)$ is a set system determining a sharing-based construction for τ-TPE in which the sets $S(\cdot)$ are defined for every predicate and attribute. We also define new sets $S(a, f) = S(f)$ for every $(a, f) \in \mathbb{A} \times \mathbb{F}$. By definition of the OR-based construction from IBE, we note that $S(a_*) \cap S(f_*) \neq \emptyset$ (where we identify the identities and the corresponding decryption keys). This way, it is easy to see that the sampling algorithm of Lemma 3.5 directly violates the security of the original sharing-based construction by giving a valid decryption key in the set $\cup_i S(a_i, f_i)$. □

3.4 The Communication Complexity Approach

In this section we show an alternative approach to refute sharing-based constructions of predicate encryption schemes using separation results in two-party communication complexity. The strength of this new approach is its generality. In particular, using conjectured separations in communication complexity, we prove the impossibility of a sharing-based construction of $\mathbf{NC}^1$-PE from $\mathbf{AC}^0$-PE, thus making some progress toward the question of separating PE schemes based on the complexity classes the underlying predicates come from. On the other hand, we are currently able to apply this approach only to sharing-based constructions rather than to general black-box constructions. In contrast, Lemma 3.5 plays a critical role in our proof of full black-box separation of TPE from IBE. Lemma 3.5 indeed proves a stronger result than needed to refute sharing-based constructions from IBE as applied in Corollary 3.10. In particular, we are able to handle the sets $S(a, f)$ on the edges of the bipartite graph given by $(\mathbb{A}, \mathbb{F})$. These sets naturally arise in the attack strategy we employ in our proof.

Let $(\mathbb{A}, \mathbb{F})$ be a predicate encryption scheme. W.l.o.g. we identify $\mathbb{A}$ with $\{0,1\}^\kappa$ and think of $\mathbb{F}$ as a family of functions $\{f_b : \{0,1\}^\kappa \to \{0,1\}\}_{b\in\{0,1\}^\kappa}$, i.e., we assume for simplicity that $|\mathbb{F}| = 2^\kappa$ and its members are also indexed by $b \in \{0,1\}^\kappa$. We may sometimes abuse this notation a bit and refer to b itself as a member of $\mathbb{F}$. We can then talk about the communications complexity of $\mathbb{F}$ when $b \in \mathbb{F}$ is given to Bob and $a \in \mathbb{A}$ to Alice. As usual, we can represent this communication complexity problem by the $\{0,1\}$-matrix with rows indexed by $\mathbb{A}$ and columns by $\mathbb{F}$. With a little more abuse of notation, we denote this matrix also by $\mathbb{F} = (f_b(a))_{a,b}$ and refer to the communication complexity of $\mathbb{F}$. Recall that the essential resource in communication complexity is the number of bits Alice and Bob need to communicate to determine $f_b(a)$. Various models such as deterministic, randomized (public or private coins), nondeterministic, etc., communication complexity can be defined naturally. For details on such models, we refer to the classic book by Kushilevitz and Nisan [24], the paper by Babai et al. [2], and the surveys by Lokam [28] and Lee and Shraibman [25].

To connect communication complexity to OR-based constructions using IBE, we use the model of Merlin-Arthur games in communication complexity:

Definition 3.11 (Merlin-Arthur Protocols in Communication Complexity)**.** A matrix $\mathbb{F}$ is said to have an MA-protocol of complexity $\ell + c$ if there exists a c-bit randomized public-coin verification protocol Π between Alice and Bob such that

- $\mathbb{F}(a, b) = 1 \Rightarrow \exists w \in \{0,1\}^\ell \quad \Pr[\Pi((a, w), (b, w)) = 1] \geq 2/3$,
- $\mathbb{F}(a, b) = 0 \Rightarrow \forall w \in \{0,1\}^\ell \quad \Pr[\Pi((a, w), (b, w)) = 1] \leq 1/3$.

The MA-complexity of $\mathbb{F}$, denoted $\mathsf{MA}(\mathbb{F})$, is the minimum complexity of an MA protocol for the matrix $\mathbb{F}$.

With this definition, the well-known fact (see, for example, [24]) that $\mathsf{EQUALITY}$ has public coin randomized communication complexity of $O(1)$, and our Definition 3.2 of OR-construction, the following lemma is easy.

Lemma 3.12. *Suppose there is an* OR*-based construction of a predicate encryption scheme* $(\mathbb{A}, \mathbb{F})$ *using an IBE scheme* $(\mathbb{B}, \mathbb{G})$*. Then* $\mathsf{MA}(\mathbb{F}) = O(\log \kappa)$*.*

Proof. Merlin's proof (a nondeterministic move) w can simply be the label of an input wire to the OR gate, namely, $\vee_{i,j}\varphi(\alpha_j)$. Since $(\mathbb{B}, \mathbb{G})$ is an IBE scheme, $\varphi_i(\alpha_j) = 1$ iff $\varphi_i = \alpha_j$, i.e., each of the inputs to the OR gate is an EQUALITY function. Alice and Bob can run an $O(1)$-bit randomized protocol to test this equality. This protocol is Arthur's verification protocol. Since i and j are bounded by poly(κ), it follows that the length of Merlin's proof is $\ell = O(\log \kappa)$, because w is simply consisting of two indices i and j in the sets assigned to f and a. Thus $\mathsf{MA}(\mathbb{F}) = O(\log \kappa)$. □

The following result due to Klauck [23] gives a matrix with high MA communication complexity:

Theorem 3.13 (Klauck)**.** *Let DISJ denote the disjointness matrix on sets in a universe of size κ, i.e., $\mathsf{DISJ}(x, y) = 1$ if and only if $x \cap y = \emptyset$ for $x, y \subseteq [\kappa]$. Then $\mathsf{MA}(\mathsf{DISJ}) = \Omega(\sqrt{\kappa})$. This, in fact, holds even when $|x| = |y| = \epsilon\kappa$ for any constant ϵ with $\frac{1}{4} \leq \epsilon < \frac{1}{2}$.*

Theorem 3.14. *For some constant $0 < \tau < 1$, e.g., $\tau = 1/3$, there is no OR-based (and hence no sharing-based) construction of a τ-TPE scheme from IBE.*

Proof of Lemma 3.14. By Lemma 3.12, if $(\mathbb{A}, \mathbb{F})$ has an OR-based construction from an IBE, then the matrix $\mathbb{F}$ has MA-complexity at most polylog(κ). On the other hand, we will show that if $(\mathbb{A}, \mathbb{F})$ defines a τ-TPE scheme with $\tau = 1/3$, then $\mathbb{F}$ must have MA-complexity $\Omega(\sqrt{\kappa})$. Recall from Definition 2.7 that $\mathbb{F}(a, b) = 1$ if and only if $\langle a, b \rangle \geq \kappa/3$. As in the proof of Lemma 3.5, we consider the set of vectors $\mathcal{A} \subseteq \mathbb{A}$ of Hamming weight $\kappa/3$ and $\mathcal{F} \subseteq \mathbb{F}$ of Hamming weight $2\kappa/3$ and threshold $1/3$. Then it is clear that $\mathbb{F}(a, b) = 1$ for $a \in \mathcal{A}$ and $b \in \mathcal{F}$ if and only if, as sets, $a \subseteq b$, i.e., $a \cap \bar{b} = \emptyset$. Thus let $x, y \subseteq [\kappa]$, $|x| = |y| = \kappa/3$ be an input for DISJ. Define a to be the characteristic vector of x and b to be that of $\bar{b}$. It is then easy to see that $\langle a, b \rangle \geq \kappa/3$ if and only if $x \cap y = \emptyset$. □

To derive separations among stronger predicate encryption schemes based on sharing constructions, we need to recall definitions of languages and complexity classes in two-party communication complexity, in particular, $\mathbf{PH}^{\mathrm{cc}}$ and $\mathbf{PSPACE}^{\mathrm{cc}}$.

Complexity classes in two-party communication complexity are defined in terms of languages consisting of pairs of strings (a, b) such that $|a| = |b|$. Denote by $\{0,1\}^{2*}$ the universe $\{(a, b) : a, b \in \{0,1\}^* \text{ and } |a| = |b|\}$. For a language $L \subseteq \{0,1\}^{2*}$, we denote its characteristic function on pairs of strings of length κ by L_κ. L_κ is naturally represented as an $2^\kappa \times 2^\kappa$ matrix with $\{0,1\}$ or ± 1 entries.

Conversely, if $A = \{A_K\}$ is an infinite sequence of $\{0,1\}$-matrices (where A_K is $K \times K$), then we can associate a language L_A with A and talk about its communication complexity. L_A is not necessarily unique (since the K's may be different from powers of two), but for the purposes of lower bounds we will fix one such language and refer to it as *the* language L_A corresponding to A.

Definition 3.15. Let $l_1(\kappa), \ldots, l_d(\kappa)$ be nonnegative integers such that $l(\kappa) := \sum_{i=1}^{d} l_i(\kappa) \leq (\log \kappa)^c$ for a fixed constant $c \geq 0$.

A language $L \subseteq \{0,1\}^{2*}$ is in $\mathbf{\Sigma}_d^{\mathrm{cc}}$ if there exist $l_1(\kappa), \ldots, l_d(\kappa)$ as above and Boolean functions $\varphi, \psi : \{0,1\}^{\kappa + l(\kappa)} \longrightarrow \{0,1\}$ such that $(a, b) \in L_\kappa$ if and only if

$$\exists u_1 \, \forall u_2 \ldots Q_d u_d \, (\varphi(a, u) \diamondsuit \psi(b, u)),$$

where $|u_i| = l_i(\kappa), u = u_1 \ldots u_d$, Q_d is $\forall$ for d even and is $\exists$ for d odd, and, $\diamondsuit$ stands for $\vee$ if d is even and for $\wedge$ if d is odd.

Definition 3.16.

- By allowing a bounded number of alternating quantifiers in Definition 3.15, we get an analog of the polynomial time hierarchy: $\mathbf{PH}^{\mathrm{cc}} = \bigcup_{d\geq 0} \mathbf{\Sigma}_d^{\mathrm{cc}}$.
- We get an analog of **PSPACE**, by allowing an unbounded, but no more than polylog(κ), number of alternating quantifiers in Definition 3.15 : $\mathbf{PSPACE}^{\mathrm{cc}} = \bigcup_{c>0} \bigcup_{d\leq(\log\kappa)^c} \mathbf{\Sigma}_d^{\mathrm{cc}}$.

The following lemma shows a connection between the communication complexity class $\mathbf{PH}^{\mathrm{cc}}$ and OR-based constructions using $\mathbf{AC}^0$-predicate encryption.

Lemma 3.17. *Suppose a predicate encryption scheme* $(\mathbb{A}, \mathbb{F})$ *is obtained by an* OR*-based construction using an* $\mathbf{AC}^0$*-predicate encryption scheme. Then the language given by the sequence of matrices* $\{\mathbb{F}\}_\kappa$ *is in* $\mathbf{PH}^{\mathrm{cc}}$.

Proof of Lemma 3.17. By hypothesis, for a given $f_b \in \mathbb{F}$, we have $\mathbf{AC}^0$ circuits $\varphi_{1b}, \ldots, \varphi_{qb}$ and for a given $a \in \mathbb{A}$, we have $\alpha_{1a}, \ldots, \alpha_{qa}$ such that $f_b(a) = \vee_{i,j}\varphi_{ib}(\alpha_{ja})$. Knowing f_b, Bob can compute the circuit

$$C_y(z) \equiv \bigvee_{ij} \varphi_{iy}(z_j), \text{ where } z = (z_1, \ldots, z_q), \ \ |z_j| = |\alpha_j|.$$

Knowing a, Alice can compute $\alpha_a = (\alpha_{1a}, \ldots, \alpha_{qa})$ on which C_b needs to be evaluated.

We give a protocol with a bounded number of alternations for $\mathbb{F}$. Let the depth of C_b be d (including the top OR-gate). An existential player will have a move for an OR gate in C_b and a universal player will have a move for an AND gate. Their d moves will describe an accepting path in C_b on α_a. For example, assuming AND and OR gates alternate in successive layers, $\exists w_1 \forall w_2 \cdots Q_d w_d \gamma(C_b, w_1, \ldots, w_d)(\alpha_a)$ describes a path in C_b – start with the top OR gate and follow the wire w_1 to the AND gate below and then the wire w_2 from this gate and so on – ending in a gate $\gamma := \gamma(\ldots)$ to witness the claim that $f_b(a) = 1$. Since Bob knows C_b, he can verify the correctness of the path $w_1 w_2 \cdots w_k$ in the circuit and the type of the gate γ given by the path. He then sends the labels of the inputs and the type (AND or OR) of the gate to Alice, who responds with $\gamma(\alpha_a)$. Bob can verify that this will ensure $C_b(\alpha_a) = 1$. On the other hand, if $C_b(\alpha_a) = 0$, then it is easy to see that the existential player will not have a winning strategy to pass verification protocol of Alice and Bob on their inputs a and C_b. It follows that $\mathbb{F}$ has a protocol with at most d alternations and hence $\{\mathbb{F}\}_\kappa \in \mathbf{PH}^{\mathrm{cc}}$. □

This lemma enables us to show the impossibility of OR-based (hence sharing-based) constructions of predicate encryption schemes using $\mathbf{AC}^0$-predicate encryption. In particular, if we can separate $\mathbf{PSPACE}^{\mathrm{cc}}$ from $\mathbf{PH}^{\mathrm{cc}}$, we can show that there's no sharing-based construction of $\mathbf{NC}^1$-predicate encryption using $\mathbf{AC}^0$-predicate encryption. However, it is a longstanding open question in communication complexity to separate $\mathbf{PSPACE}^{\mathrm{cc}}$ from $\mathbf{PH}^{\mathrm{cc}}$. Currently it is known [27, 32] that such a separation holds if certain Boolean matrices can be shown to have high rigidity, a connection we explain in Appendix **??**.

Theorem 3.18.

1. *Suppose* $\mathbf{PH}^{\mathrm{cc}} \neq \mathbf{PSPACE}^{\mathrm{cc}}$. *Then, there is no* OR*-based construction of an* $\mathbf{NC}^1$*-PE scheme from any* $\mathbf{AC}^0$*-PE scheme. In particular, there is an* $\mathbf{NC}^1$*-function family* $\mathbb{F}$ *derived from the Sipser functions such that* $(\mathbb{A}, \mathbb{F})$ *does not have an* OR*-based construction from any* $\mathbf{AC}^0$*-predicate encryption scheme.*

2. *Suppose Hadamard matrices are as highly rigid as demanded in Theorem* **??**. *Then there is no* OR*-based construction of an* $\mathbf{NC}^1$*-PE scheme from any* $\mathbf{AC}^0$*-PE scheme. In particular, predicate encryption defined by the Inner Product mod 2 function does not have an* OR*-based construction from any* $\mathbf{AC}^0$*-predicate encryption scheme.*

Proof. Suppose $\mathbf{PH}^{\mathrm{cc}} \neq \mathbf{PSPACE}^{\mathrm{cc}}$. Let $\psi : \{0,1\}^\kappa \times \{0,1\}^\kappa \to \{0,1\}$ be the following "Sipser function" of depth $\log 2\kappa$: it is a complete binary tree with alternating layers of AND and OR gates and each of the bottom gates receives as inputs an a_i and a b_i for $1 \leq i \leq \kappa$. Computing $\psi(a,b)$ when Alice holds $a \in \{0,1\}^\kappa$ and Bob holds $b \in \{0,1\}^\kappa$ is easily seen to be a $\mathbf{PSPACE}^{\mathrm{cc}}$-complete problem. Hence it is not $\mathbf{PH}^{\mathrm{cc}}$. It is also obvious that for any fixed b, $\psi(\cdot, b)$ is an $\mathbf{NC}^1$ function. Defining $\mathbb{A} = \{0,1\}^\kappa$ and $\mathbb{F} = \{\psi(\cdot, b)\}_b$, we see that $(\mathbb{A}, \mathbb{F})$ is an $\mathbf{NC}^1$-PE problem. Since the language given by $\mathbb{F}$ is not in $\mathbf{PH}^{\mathrm{cc}}$, by Lemma 3.17, we conclude that $(\mathbb{A}, \mathbb{F})$ does not have an OR-based construction from any $\mathbf{AC}^0$-predicate encryption.

Consider the Inner Product mod 2 function: $\mathsf{ip2}(a,b) := \sum_{i=1}^{\kappa} a_i b_i \bmod 2$ for $(a,b) \in \{0,1\}^\kappa \times \{0,1\}^\kappa$. The corresponding $\{-1,+1\}$-matrix $H = ((-1)^{\mathsf{ip2(a,b)}})_{a,b}$ is well-known to be an Hadamard matrix. Thus if Hadamard matrices are sufficiently rigid to apply Theorem **??**, we can conclude that the communication problem given by $\mathsf{ip2}$ is not in $\mathbf{PH}^{\mathrm{cc}}$. Now, the proof of the second part is similar to the above. □

4 Separating TPE from IBE

In this section we prove that there is no general black-box construction of threshold predicate encryption schemes from identity-based encryption schemes.

Theorem 4.1. *Let $\kappa \in \mathbb{N}$ be the security parameter. Then, there exists an oracle $\mathcal{O}$ relative to which CCA secure IBE schemes exist, as per the Definition 2.5. However, for any constant $0 < \tau < 1$, there exists a query-efficient (i.e., that makes at most* $\mathrm{poly}(\kappa)$ *queries to $\mathcal{O}$) adversary* **Adv** *that can break even the* CPA security *of any τ-TPE scheme relative to $\mathcal{O}$, again as per the Definition 2.5. Moreover,* **Adv** *can be implemented in* $\mathrm{poly}(\kappa)$*-time if given access to a* **PSPACE** *oracle, and its success probability can be made arbitrarily close to the completeness of the τ-TPE scheme.*

In the next three subsections, we prove this theorem. Even though our overall proof strategy is somewhat similar to that of [11], there are several conceptual and technical differences as explained in the introduction. In Section 4.1, we define our random IBE oracle, prove some of its properties, and define some operations that allow us to assume a nice structure for such an oracle. In Section 4.2 we describe our attack on a τ-**TPE** relative to this IBE oracle. Finally, we analyze this attack for query efficiency and success probability in Section 4.3. Lemma 4.13 gives a lower bound on the success probability of the attack. Its proof involves three experiments and estimating the statistical distance between successive pairs in them. The analysis also uses the combinatorial lemma 3.5.

4.1 The Oracle

We begin the proof of Theorem 4.1 by first defining our random IBE oracle, $\mathcal{O}_{\mathbf{IBE}}$, also denoted by $\mathcal{O}$ for short, (which trivially implies a CCA secure IBE as outlined in Remark 4.3), and then breaking any τ-TPE (with a constant τ) relative to this oracle.

Construction 4.2 (The randomized oracle $\mathcal{O} = (\mathbf{g}, \mathbf{k}, \mathbf{id}, \mathbf{e}, \mathbf{d})$)**.** By $\mathcal{O}_\lambda$ we refer to the part of $\mathcal{O}$ whose answers are λ bits, and $\mathcal{O}$ is the union of $\mathcal{O}_\lambda$ for all λ.

- The $\mathsf{master\text{-}key\ generating\ oracle}$ $\mathbf{g} : \{0,1\}^\lambda \mapsto \{0,1\}^\lambda$ is a random permutation that takes as input a secret-key $\mathsf{sk} \in \{0,1\}^\lambda$, and returns a public-key $\mathsf{pk} \in \{0,1\}^\lambda$.

- The $\mathsf{decryption\text{-}key\ generating\ oracle}$ $\mathbf{k} : \{0,1\}^{2\lambda} \mapsto \{0,1\}^\lambda$ takes as input a secret-key $\mathsf{sk} \in \{0,1\}^\lambda$ and an identity $\alpha \in \{0,1\}^\lambda$, and returns a decryption-key $\mathsf{dk}_\alpha \in \{0,1\}^\lambda$. We require $\mathbf{k}(\mathsf{sk}, \cdot)$ to be a random permutation over $\{0,1\}^\lambda$ for every $\mathsf{sk} \in \{0,1\}^\lambda$.

- The $\mathsf{identity\ finding\ oracle}$ $\mathbf{id} : \{0,1\}^{2\lambda} \mapsto \{0,1\}^\lambda$ takes as input a public-key $\mathsf{pk} \in \{0,1\}^\lambda$ and a decryption-key $\mathsf{dk} \in \{0,1\}^\lambda$, and returns the unique α such that $\mathbf{k}(\mathsf{sk}, \alpha) = \mathsf{dk}$, where $\mathsf{sk} = \mathbf{g}^{-1}(\mathsf{pk})$.

- The $\mathsf{encryption\ oracle}$ $\mathbf{e} : \{0,1\}^{3\lambda} \mapsto \{0,1\}^\lambda$ takes as input a public-key $\mathsf{pk} \in \{0,1\}^\lambda$, an identity $\alpha \in \{0,1\}^\lambda$ and a message $m \in \{0,1\}^\lambda$, and returns a ciphertext $c \in \{0,1\}^\lambda$. We require $\mathbf{e}(\mathsf{pk}, \alpha, \cdot)$ to be a random permutation over $\{0,1\}^\lambda$ for every $(\mathsf{pk}, \alpha) \in \{0,1\}^{2\lambda}$.

- The $\mathsf{decryption\ oracle}$ $\mathbf{d} : \{0,1\}^{3\lambda} \mapsto \{0,1\}^\lambda$ takes as input a public-key $\mathsf{pk} \in \{0,1\}^\lambda$, a decryption-key $\mathsf{dk} \in \{0,1\}^\lambda$ and a ciphertext $c \in \{0,1\}^\lambda$, and returns the unique m such that $\mathbf{e}(\mathsf{pk}, \alpha, m) = c$, where $\alpha = \mathbf{id}(\mathsf{pk}, \mathsf{dk})$.

By an IBE oracle, we refer to an oracle in the support set of $\mathcal{O}$, $\mathrm{Supp}(\mathcal{O})$, and by a partial IBE oracle we refer to a partial oracle that could be extended to an oracle in $\mathrm{Supp}(\mathcal{O})$.

Remark 4.3 (CCA secure IBE relative to $\mathcal{O}$)**.** To encrypt a bit $b \in \{0,1\}$ under the identity α and the public-key pk, the encryption algorithm will randomly extend b to a λ-bit random string: $m = (b, b_1, \ldots, b_{\lambda-1}), (b_1, \ldots, b_{\lambda-1}) \xleftarrow{\$} \{0,1\}^{\lambda-1}$ and gets the encryption $c = \mathbf{e}(\mathsf{pk}, \alpha, m)$. To decrypt, one first decrypts c and then take the first bit as the encrypted bit. By independently encrypting the bits of an input message $m = (m_1, \ldots, m_n)$ of length $n = \mathrm{poly}(\kappa)$, and by using a standard hybrid argument, one can generalize the scheme to arbitrary long messages. The construction that we briefly described above is only CPA secure, where any adversary has advantage at most $2^{-\Theta(\kappa)}$. But, this can easily and in a black-box manner be transformed into a CCA secure construction, without incurring any additional assumptions, using the Fujisaki-Okamoto transform [15,16] in the random oracle model [6]. We note that even though $\mathcal{O}$ is not exactly a random oracle, for our purposes it suffices to use one of the sub-oracles of $\mathcal{O}$ as a random oracle in the above transform.

The sub-oracle $\mathbf{id}$ is not required to get a secure IBE scheme, it will be used by our query-efficient adversary who breaks any τ-TPE scheme (for constant τ) relative to $\mathcal{O}$. We also note that adding this sub-oracle to $\mathcal{O}$ does not prevent $\mathcal{O}$ from realizing a *secure* IBE scheme. It is easy to see that in the IBE security game (as a special case of the security game of predicate encryption), for every decryption-key that he gets from the challenger, the adversary already knows the corresponding identities.

As we will see later, any τ-TPE scheme (for constant τ) relative to $\mathcal{O}$ can be broken by a $\mathrm{poly}(\kappa)$-query adversary which suffices to get a black-box separation of τ-TPE from IBE. Since our adversary can be easily implemented efficiently given access to a **PSPACE**-complete oracle, it also shows that no relativizing reduction from τ-TPE to IBE) exists either [20][2].

To describe and analyze our attack, we first need to formally define and study some properties of our random IBE oracle $\mathcal{O}$.

[2] A good "approximation" of the attack can also be implemented assuming $\mathbf{P} = \mathbf{NP}$.

Definition 4.4 (Root, Dual, Triangle)**.** Let $\mathcal{O} = (\mathbf{g}, \mathbf{k}, \mathbf{id}, \mathbf{e}, \mathbf{d})$ be the (randomized) oracle of Construction 4.2.

- For any $(\mathsf{sk}, \mathsf{pk}, \alpha, \mathsf{dk}) \in \{0,1\}^{4\lambda}$, if $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$ and $\mathbf{k}(\mathsf{sk}, \alpha) = \mathsf{dk}$, then we call $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$ the *root* query; $\mathbf{k}(\mathsf{sk}, \alpha) = \mathsf{dk}$ and $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$ the *dual* (to each other) queries; and all three of them the *triangle*.
- For any $(\mathsf{pk}, \alpha, \mathsf{dk}, m, c) \in \{0,1\}^{5\lambda}$, if $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$ and $\mathbf{e}(\mathsf{pk}, \alpha, m) = c$, then we call $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$ the *root* query; $\mathbf{e}(\mathsf{pk}, \alpha, m) = c$ and $\mathbf{d}(\mathsf{pk}, \mathsf{dk}, c) = m$ the *dual* (to each other) queries; and all three of them the *triangle*.

Definition 4.5 (Closure of Partial IBE Oracles)**.** Let $\mathcal{P} = (\mathbf{g}, \mathbf{k}, \mathbf{id}, \mathbf{e}, \mathbf{d})$ be a partial IBE oracle. By the *closure* of $\mathcal{P}$, denoted as $\overline{\mathcal{P}} = (\overline{\mathbf{g}}, \overline{\mathbf{k}}, \overline{\mathbf{id}}, \overline{\mathbf{e}}, \overline{\mathbf{d}})$ we refer to another partial oracle which contains $\mathcal{P}$ as a subset and *in addition* it might have answers defined for more queries as follows.

1. If $[\mathbf{g}(\mathsf{sk}) = \mathsf{pk}] \in \mathcal{P}$ then:
   (a) If $[\mathbf{k}(\mathsf{sk}, \alpha) = \mathsf{dk}] \in \mathcal{P}$ then add the dual query $\overline{\mathbf{id}}(\mathsf{pk}, \mathsf{dk}) = \alpha$ to $\overline{\mathcal{P}}$.
   (b) If $[\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha] \in \mathcal{P}$ then add the dual query $\overline{\mathbf{k}}(\mathsf{sk}, \alpha) = \mathsf{dk}$ to $\overline{\mathcal{P}}$.
2. If $[\overline{\mathbf{id}}(\mathsf{pk}, \mathsf{dk}) = \alpha] \in \overline{\mathcal{P}}$ (even if $[\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha] \notin \mathcal{P}$) then:
   (a) If $[\mathbf{e}(\mathsf{pk}, \alpha, m) = c] \in \mathcal{P}$, then add the dual query $\overline{\mathbf{d}}(\mathsf{pk}, \mathsf{dk}, c) = m$ to $\overline{\mathcal{P}}$.
   (b) If $[\mathbf{d}(\mathsf{pk}, \mathsf{dk}, c) = m] \in \mathcal{P}$, then add the dual query $\overline{\mathbf{e}}(\mathsf{pk}, \alpha, m) = c$ to $\overline{\mathcal{P}}$.

In all the cases above, the dual query which is added to $\overline{P}$ is called the *dependent* query (with respect to $\mathcal{P}$). For any query x, if $x \notin \overline{\mathcal{P}}$ we call x a *free* query (again, with respect to $\mathcal{P}$).

Definition 4.6 (Normal Form)**.** We say an oracle algorithm $A^{\mathcal{O}}$ is in *normal form*, if:

- When A is about to ask an oracle query x of the form $\mathbf{k}(\mathsf{sk}, \alpha) = \mathsf{dk}$, then *before* asking x, it makes the query $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$, and after asking x it also makes the query $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$.
- When A is about to ask an oracle query x of the form $\mathbf{d}(\mathsf{pk}, \mathsf{dk}, c) = m$, then before asking x, it makes the query $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$, and after asking x it also makes the query $\mathbf{e}(\mathsf{pk}, \alpha, m) = c$.

By the *normalized* version of an algorithm A, we denoted the modified (but with the same functionality as before) version of it that asks some required additional queries as above, so that the resulting algorithm is in normal form. We also call a partial oracle $\mathcal{P} = (\mathbf{g}, \mathbf{k}, \mathbf{id}, \mathbf{e}, \mathbf{d})$ *normalized* if it has the following two properties:

- If $[\mathbf{k}(\mathsf{sk}, \alpha) = \mathsf{dk}] \in \mathcal{P}$, then $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$ and $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$ are also defined in $\mathcal{P}$ for some pk.
- If $[\mathbf{d}(\mathsf{pk}, \mathsf{dk}, c) = m] \in \mathcal{P}$, then $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$ and $\mathbf{e}(\mathsf{pk}, \alpha, m) = c$ are also defined in $\mathcal{P}$ for some α.

The view of any normalized algorithm A clearly contains a normalized partial oracle.

Note that by normalizing any algorithm, the number of queries that it asks increases by at most a factor of 3.

Closure of Normalized Partial Oracles. When we take the closure of a normalized partial oracle $\mathcal{P}$, a slightly simpler definition can be used. In particular, in the second bullet of Definition 4.5, we can condition at the case $[\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha] \in \mathcal{P}$ (rather than using the condition $[\overline{\mathbf{id}}(\mathsf{pk}, \mathsf{dk}) = \alpha] \in \overline{\mathcal{P}}$). This is because due to normal form property, the query/answer $\overline{\mathbf{id}}(\mathsf{pk}, \mathsf{dk}) = \alpha$ already exists in $\mathcal{P}$. This modification of the definition of closure will simplify some of our arguments in the next subsections. Note that for a normalized partial oracle $\mathcal{P}$, taking the closure can at most double the number of queries: $|\overline{\mathcal{P}}| \leq 2 \cdot |\mathcal{P}|$.[3]

The next two lemmas hold in a more general setting where normal form is not required, but the proof is easier in the case of normal form (and we only need this special case), so we only specify and prove them in the special case of normal form.

Lemma 4.7 (Lazy Evaluation)**.** *Suppose $\mathcal{P}$ is a partial IBE oracle over queries of output length λ (i.e., there exists $\mathcal{Q} \in \operatorname{Supp}(\mathcal{O}_\lambda)$ such that $\mathcal{P} \subseteq \mathcal{Q}$), and let $|\mathcal{P}| = t$ (i.e., there are t queries defined in $\mathcal{P}$) and $t < 2^{\lambda-1}$. Then, for any query x to $\mathcal{O}$ which is not inside the closure of $\mathcal{P}$ (i.e., $x \notin \overline{\mathcal{P}}$), the statistical distance between the answer of $\mathcal{O}$ conditioned on $\mathcal{P}$ being part of $\mathcal{O}$ (i.e., $(\mathcal{O}(x) \mid \mathcal{P} \subseteq \mathcal{O})$) and the uniform distribution over $\{0,1\}^\lambda$ is at most $O(t/2^\lambda)$. (Note that the answers to the queries in $\overline{\mathcal{P}}$ are already fixed.)*

Proof of Lemma 4.7. We can assume w.l.o.g. that $\mathcal{P} = \overline{\mathcal{P}}$ (because otherwise we can let $\mathcal{Q} = \overline{\mathcal{P}}$ and work with $\mathcal{Q}$ instead of $\mathcal{P}$). We call such partial oracles $\mathcal{P} = \overline{\mathcal{P}}$ *self-closured.* We also simply assume that t is still the size of $\mathcal{P}$ (even though its value might change by a factor of 2 after moving to the closure of $\mathcal{P}$).

The proof has the following general steps: **(1)** For a self-closured $\mathcal{P}$ we present a randomized procedure that samples a full oracle from the distribution $(\mathcal{O}_\lambda \mid \mathcal{P} \subseteq \mathcal{O}_\lambda)$. **(2)** Then we will show that during the above randomized sampling process, the answer to any query x such that $x \notin \mathcal{P}$, is chosen $O(t/2^\lambda)$-close to uniform over $\{0,1\}^\lambda$.

For starters suppose the IBE oracles does *not* have any encryption $\mathbf{e}$ or decryption $\mathbf{d}$ part and only consists of $(\mathbf{g}, \mathbf{k}, \mathbf{id})$. A crucial point is that for any $(\mathsf{sk}_1, \mathsf{pk}_1) \neq (\mathsf{sk}_2, \mathsf{pk}_2) \in \{0,1\}^{2\lambda}$, if $[\mathbf{g}(\mathsf{sk}_1) = \mathsf{pk}_1] \in \mathcal{O}$ and $[\mathbf{g}(\mathsf{sk}_2) = \mathsf{pk}_2] \in \mathcal{O}$, then the queries to $\mathbf{k}$ and $\mathbf{id}$ sub-oracles that are related to $(\mathsf{sk}_1, \mathsf{pk}_1)$ (i.e., their first input is either of sk_1 or pk_1) are answered independently of the queries that are related to $(\mathsf{sk}_2, \mathsf{pk}_2)$. Based on (variants of) this simple observation (and relying on the fact that $\mathcal{P}$ is a normal self-closured partial oracle of size t), in the simplified case of IBE oracles (without $\mathbf{e}$ and $\mathbf{d}$ sub-oracles), we can extend the partial oracle $\mathcal{P}$ to a full oracle as follows:

1. For every query $[\mathbf{g}(\mathsf{sk}) = \mathsf{pk}] \in \mathcal{P}$, we finish sampling the queries (and their responses) to $\mathbf{k}$ and $\mathbf{id}$ sub-oracles that are related to (either of) $(\mathsf{sk}, \mathsf{pk})$. Note that since $\mathcal{P}$ is self-closured, for every query to $\mathbf{k}$ or $\mathbf{id}$ that involves (either of) $(\mathsf{sk}, \mathsf{pk})$, its dual is also already defined in $\mathcal{P}$. So, all we have to do is to randomly match the remaining queries that involve $(\mathsf{sk}, \mathsf{pk})$ in dual pairs. The answers that are sampled in this step are chosen uniformly at random from a space of size at least $2^\lambda - t$, and so the answers are $O(t/2^\lambda)$-close to uniform. After this step all the $\mathbf{k}$ and $\mathbf{id}$ queries that could be related to $(\mathsf{sk}, \mathsf{pk})$ are defined.

   At this point we say that we have *covered* all the $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$ queries inside $\mathcal{P}$. We also say that a $\mathbf{k}$ or $\mathbf{id}$ query inside $\mathcal{P}$ is covered in this step, if they are related to some $(\mathsf{sk}, \mathsf{pk})$ such that $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$ is covered. Note that all the $\mathbf{k}$ queries of $\mathcal{P}$ will be covered in the step for sure, because due to the normal form property of $\mathcal{P}$ all of the $\mathbf{k}(\mathsf{sk}, \cdot)$ queries make the relevant $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$ query to be asked (and exist in $\mathcal{P}$) as well.

[3] For general partial oracles $\mathcal{P}$, it holds that $|\overline{\mathcal{P}}| \leq O(|\mathcal{P}|)$.

2. In this step we go over the remaining uncovered queries of $\mathcal{P}$ which could only be some $\mathbf{id}$ queries. For any such query $[\mathbf{id}(\mathsf{pk},\mathsf{dk}) = \alpha] \in \mathcal{P}$, we would like to sample some $\mathsf{sk} \in \{0,1\}^\lambda$ and set $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$, but we want to do it from its correct distribution conditioned on the sampled parts of $\mathcal{O}$ and $\mathcal{P}$. It is easy to see that the distribution of sk in this case is completely uniform among all the sk's such that the query $\mathbf{g}(\mathsf{sk})$ is not already covered.

3. After sampling some sk and pairing them with some pk's for all the queries of the form $[\mathbf{id}(\mathsf{pk},\mathsf{dk}) = \alpha] \in \mathcal{P}$ in the previous step, now we can go over these newly generated mappings $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$, and cover their related $\mathbf{k}$ and $\mathbf{id}$ queries as in Step 1.

4. Then, we go over all the unused pk's and sk's and randomly pair them together to complete the description of the sub-oracle $\mathbf{g}$. In this step also, the answers are chosen from a space that is of size at least $2^\lambda - t$.

5. For any mapping $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$ generated in the previous step, we choose a random permutation over $\{0,1\}^\lambda$ and use that permutation to define the relevant $\mathbf{k}$ and $\mathbf{id}$ queries.

If we go over the above steps, it can be seen that the *only* step that the sampled answers might *not* be chosen from a space of size at least $2^\lambda - t$ is in Step 2. In particular, suppose sk is selected in this step and is assigned to some already fixed pk through the query $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$. The key point is that for any such fixed query $\mathbf{g}(\mathsf{sk})$ *out* of $\mathcal{P}$ the probability that it is chosen to be mapped to some fixed pk is at most $t/2^\lambda$, simply because the number of such pk's is at most t. On the other hand, if sk is not selected in Step 2, its answer will be chosen in other steps which will be from a space of size at least $2^\lambda - t$. Therefore, the statistical distance between the sampled answer to $\mathbf{g}(\mathsf{sk})$ and the uniform over $\{0,1\}^\lambda$ will be at most $t/2^\lambda + O(t/2^\lambda)$ which is still $O(t/2^\lambda)$.

For the general case of IBE oracles, we follow a similar sampling procedure and then will study the statistical distance between the sampled answers and the uniform distribution over $\{0,1\}^\lambda$. The sampling has more steps, and we choose to write it more concisely (some of the steps corresponding to the simplified case above will be merged into one step), but the main ideas are very similar to the simpler case of IBE oracles studied above.

1. For every query $[\mathbf{id}(\mathsf{pk},\mathsf{dk}) = \alpha] \in \mathcal{P}$, consider all the $\mathbf{e}$ and $\mathbf{d}$ queries that potentially can be related to $(\mathsf{pk},\mathsf{dk},\alpha)$ (i.e., they are of the form $\mathbf{e}(\mathsf{pk},\alpha,\cdot)$ or $\mathbf{d}(\mathsf{pk},\mathsf{dk},\cdot)$). Since $\mathcal{P}$ is self-closured, the subset of these queries that are inside $\mathcal{P}$ can be partitioned into dual pairs. So, we go over all the new $\mathbf{e},\mathbf{d}$ queries (out of $\mathcal{P}$) that relate to $(\mathsf{pk},\mathsf{dk},\alpha)$ and randomly partition them into dual pairs of queries. (Similar to the simplified case above, due to normal form property of $\mathcal{P}$, after this step there will be no $\mathbf{d}$ query in $\mathcal{P}$ that is not covered.

2. We go over all the queries of the form $[\mathsf{e}(\mathsf{pk},\alpha,m) = c] \in \mathcal{P}$ that are not covered in the previous step. If no sk is mapped to this pk we sample a random sk (from the ones that are not used yet) and add $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$ to the set of sampled queries. Then we finish sampling the $\mathbf{k}$ and $\mathbf{id}$ queries that are related to the pair $(\mathsf{sk},\mathsf{pk})$. A crucial point is that at the end of this step, we sample at most t new dk's and assign them to some $(\mathsf{sk},\mathsf{pk})$ pair such that $[\mathsf{e}(\mathsf{pk},\alpha,m) = c] \in \mathcal{P}$. Moreover, these (at most) t sampled dk's are chosen uniformly from a set of size at least $2^\lambda - t$. Therefore, for any particular fixed dk, the probability that we choose it and sample the query $\mathbf{id}(\mathsf{pk},\mathsf{dk}) = \alpha$ (while $[\mathsf{e}(\mathsf{pk},\alpha,m) = c] \in \mathcal{P}$) is at most $t/(2^\lambda - t)$ which is still $O(t/2^\lambda)$ for $t < 2^\lambda/2$.

3. For every query $[\mathbf{e}(\mathsf{pk}, \alpha, m) = c] \in \mathcal{P}$ handled in the previous step, let dk be the sampled decryption key such that $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$ is sampled there. Now, we go back to finish covering all the queries that could be related to $(\mathsf{pk}, \mathsf{dk}, \alpha)$ (similar to Step 1). An important point is that in this step, the $\mathbf{d}$ queries that are covered and are of the form $\mathbf{d}(\mathsf{pk}, \mathsf{dk}, c)$ might *not* be answered from a close-to-uniform distribution, and their answers might be already fixed to to some existing uncovered $\mathbf{e}(\mathsf{pk}, \alpha, m) = c$ query of $\mathcal{P}$. However, for any fixed such query $\mathbf{d}(\mathsf{pk}, \mathsf{dk}, c)$, as we discussed at the end of the previous step, the probability that this dk is sampled in some query $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$, is at most $O(t/2^\lambda)$, and thus its answer is still chosen $O(t/2^\lambda)$-close to uniform.

   The remaining three steps are similar to the sampling procedure of the simplified IBE oracles.

4. For every query $[\mathbf{g}(\mathsf{sk}) = \mathsf{pk}] \in \mathcal{P}$ that is not covered in the previous step, go over all of its $\mathbf{k}$ and $\mathbf{id}$ related queries and sample all of them that have not been sampled yet. At the end of this step, there is no $\mathbf{k}$ query inside $\mathcal{P}$ which is not covered yet.

5. For every uncovered query $[\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha] \in \mathcal{P}$, sample an unused sk, and add the $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$ to the set of sampled queries. Then, go over all the $\mathbf{k}$ and $\mathbf{id}$ queries that are related to $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$ and randomly partition the remaining queries into dual pairs. Then, for all of the queries of the form $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$ covered in this step (which are in or out of $\mathcal{P}$) generate a random permutation and use this permutation to define the $\mathbf{e}, \mathbf{d}$ queries related to $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$.

6. Randomly pair all the remaining sk and pk's to finish sampling of $\mathbf{g}$. Then, for each new sampled query $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$, generate a random permutation over $\{0,1\}^\lambda$ and use it to define the $\mathbf{k}$ and $\mathbf{id}$ queries that are related to $(\mathsf{sk}, \mathsf{pk})$. Finally, for each sampled query $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$ of this step, generate a random permutation over $\{0,1\}^\lambda$ and use it to define the $\mathbf{e}$ and $\mathbf{d}$ queries that are related to $(\mathsf{pk}, \mathsf{dk}, \alpha)$. This finishes the sampling of the full oracle $\mathcal{O}$ by extending $\mathcal{P}$.

In the sampling procedure above, the only steps in which some of the sampled answers might *not* be chosen from a space of size at least $2^\lambda - t$ are Steps 3 and 5. We already explained why Step 3 does not violate our claim of Lemma 4.7, and Step 5 is identical to the case that we studied in the simplified case earlier. □

Below is an important remark that will be crucial in many of the proofs.

Remark 4.8. Suppose A is a normalized algorithm asking t oracle queries to $\mathcal{O}_\lambda$. Now, instead of feeding $\mathcal{O}_\lambda$ to A, suppose we do the following: whenever a new query is in the closure of the A's currnet view's set of queries, we use the determined answer, otherwise we use a uniformly random answer from $\{0,1\}^\lambda$. Lemma 4.7 shows that this "lazy evaluation" of the oracle $\mathcal{O}_\lambda$ is $\sum_{i \in [t]} O(i/2^\lambda) = O(t^2/2^\lambda)$-close to the experiment of executing A using $\mathcal{O}_\lambda$.

Lemma 4.9 (Malformed Triangles)**.** *Suppose A is a normalized algorithm asking t oracle queries to our random IBE oracle $\mathcal{O}_\lambda$. Also, let there be three queries x_1, x_2, x_3 in the view of A which form a triangle with x_1 as its root. We say that the view of A has a malformed triangle (call it the event B), if the first appearance of x_1, the root of the triangle, in the view of A is not the first among $\{x_1, x_2, x_3\}$, the triangle. It holds that* $\Pr[B] \le O(t^2/2^\lambda)$.

Proof of Lemma 4.9. By Lemma 4.7 and Remark 4.8, if we use a lazy evaluation of the oracle $\mathcal{O}_\lambda$, then the distribution of the view of the algorithm A changes by at most a statistical distance of $O(t^2/2^\lambda)$. But when we use the lazy evaluation, at any time that a free query is asked, the probability that it becomes the root of a malformed triangle is at most $O(t/2^\lambda)$. In particular, when $\mathbf{g}(\mathsf{sk})$ is asked for the first time and is going to be the root of a malformed triangle that includes a previously asked query $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$, we would get pk as the answer to $\mathbf{g}(\mathsf{sk})$ only with probability at most $1/2^\lambda$. (Note that the previously asked node of the triangle could not be the query $\mathbf{k}(\mathsf{sk}, \alpha) = \mathsf{dk}$, because A is normalized, and therefore the query $\mathbf{k}(\mathsf{sk}, \alpha)$ would make $\mathbf{g}(\mathsf{sk})$ to be asked even before itself!).

Similarly, suppose the query $\mathbf{id}(\mathsf{pk}, \mathsf{dk})$ is being asked for the first time and is going to be the root of a malformed triangle. Similar to the argument above, the previously asked node of this triangle can not be a query of the form $\mathbf{d}(\mathsf{pk}, \mathsf{dk}, c)$, because then $\mathbf{id}(\mathsf{pk}, \mathsf{dk})$ would have already been asked from the oracle. So, suppose the previously asked node of the triangle is a query of the form $\mathbf{e}(\mathsf{pk}, \alpha, m) = c$. In this case, if $\mathbf{id}(\mathsf{pk}, \mathsf{dk})$ is a free query in the view of A at the time of being asked, then it will be responded α with probability at most $2^{-\lambda}$, and if it is due to an (inherently free) query of the form $\mathbf{k}(\mathsf{sk}, \alpha)$ (which, due to the assumed normal form property is asked right before $\mathbf{id}(\mathsf{pk}, \mathsf{dk})$), the answer to this query would be dk with probability at most $2^{-\lambda}$.

By a union bound, the probability of getting a malformed triangle in the lazy evaluation of A is at most $t^2/2^\lambda$. By incorporating the statistical distance of moving to the lazy evaluation, the probability of getting a malformed triangle remains at most $t^2/2^\lambda + O(t^2/2^\lambda) = O(t^2/2^\lambda)$. □

4.2 Our Attack

In this section we present an attack that aims to break any τ-TPE in an $\mathcal{O}$-relativized world by asking only $\mathrm{poly}(\kappa)$ queries to the random IBE oracle $\mathcal{O}$, where κ is the security parameter of the τ-TPE scheme. We prove the query-efficiency and the success probability of our attack in the next section. Similar to the attack of [11], our attack can easily be implemented in $\mathrm{poly}(\kappa)$-time if $\mathbf{P} = \mathbf{PSPACE}$, and the relativizing reductions can be ruled out by adding a **PSPACE** oracle to $\mathcal{O}$.

We first note that any black-box construction of τ-TPE scemes from IBE schemes can potentially call the oracle $\mathcal{O}_\lambda$ over different values of λ which are potentially different from the security parameter of the τ-TPE scheme itself. However, similar to [11], we assume that the τ-TPE scheme asks its queries to $\mathcal{O}_\lambda$ only for one value of λ. This assumption is purely to simplify our presentation of the attack and its analysis, and all the arguments below extend to the general case (of asking queries over any parameter $\lambda > \log s$) in a straightforward way.

We also assume that λ is large enough in the sense that $2^\lambda > s$ for an arbitrarily large $s = \mathrm{poly}(\kappa)$ that can be chosen in the description of the attack. The reason for the latter assumption is that the adversary can always ask and learn *all* the oracle queries to $\mathcal{O}$ that are of logarithmic length $O(\lambda) = O(\log \kappa)$, simply because there are at most $2^{O(\lambda)} = \mathrm{poly}(\kappa)$ many queries of this form.[4]

Now we describe our attack algorithm that participates in the security game of τ-TPE schemes in a relativized world with access to the oracle $\mathcal{O}$ of Construction 4.2. In the next section, we will show that this adversary succeeds in breaking the security with a high probability.

Construction 4.10 (**Adv** Attacking the Scheme τ-$\mathbf{TPE}^{\mathcal{O}}$)**.** The parameters are as follows. q: the total number of queries asked by the components of the scheme τ-**TPE** all together, κ: the

[4] In [11] a scheme that asks such queries is called "degenerate" and is handled similarly.

security parameter of τ-**TPE**, $\epsilon = 1/\operatorname{poly}(\kappa)$ and $s = \operatorname{poly}(\kappa)$: input parameter to the adversary **Adv**, $\lambda \leq \operatorname{poly}(\kappa)$: the parameter which determines the output length of the queries asked by the components of τ-**TPE** to the oracle $\mathcal{O}$. It is assumed that $2^\lambda > s$ for some $s = \operatorname{poly}(\kappa)$ to be chosen later.

Our adversary **Adv** executes the following.

1. **Sampling Predicates and Attributes:** **Adv** executes the sampling algorithm Samp of Lemma 3.5 with the parameter ϵ, over the predicate structure of τ-**TPE**, to get $k+1$ pairs $(a_*, f_*), \{(a_i, f_i)\}_{i\in[k]}$. Recall that this sampling is done only by knowing the predicate structure of τ-**TPE** and is independent of the actual implementation of the scheme. It can be done, for example, without the knowledge of PK.

2. **Receiving the Keys:** **Adv** receives from the challenger: the public-key PK and the decryption-keys $\{\mathsf{DK}_i\}_{i\in[k]}$, where DK_i is the generated decryption-key for f_i. We also assume that DK_* is generated by the challenger, although **Adv** does not receive it. Let V be the view of the algorithms executed by the challenger so far that generated the keys $\mathsf{PK}, \mathsf{DK}_*, \mathsf{DK}_1, \ldots, \mathsf{DK}_k$. Let $Q(V)$ be the partial oracle consisting of the queries (and their answers) specified in V. By writing in the bold font $\mathbf{V}$, we refer to V as a random variable.

3. **Encrypting Random Bits:** For all $i \in [k]$, **Adv** chooses a random bit $d \xleftarrow{\$} \{0,1\}$, computes the encryption $C_i \leftarrow \mathbf{E}(\mathsf{PK}, a_i, d)$, and then the decryption $\mathbf{D}(\mathsf{PK}, \mathsf{DK}_i, C_i)$. Let $\mathcal{L}_0$ be the partial oracle consisting of the oracle queries (and their answers) that **Adv** observes in this step.

4. **Learning Heavy Queries:** This step consists of some internal rounds. For $j = 1, 2, \ldots$ do the following. Let $\mathcal{L}_j$ be the partial oracle consisting of the oracle queries (and their answers) that **Adv** has learned about $\mathcal{O}$ till the end of the j'th round[5] of this learning step. Let $\mathbf{V}_j = (\mathbf{V} \mid \mathcal{L}_j, \mathsf{PK}, \{\mathsf{DK}\}_{i\in[k]})$ be the distribution of the random variable $\mathbf{V}$ (also including the randomness of $\mathcal{O}$) conditioned on the knowledge of $(\mathcal{L}_j, \mathsf{PK}, \{\mathsf{DK}\}_{i\in[k]})$. Now, if there is any query x such that $x \notin \mathcal{L}_j$ but $\Pr[x \in \overline{Q(\mathbf{V}_j)}] \geq \epsilon$, **Adv** asks the lexicographically first such x from the oracle $\mathcal{O}$, sets $\mathcal{L}_{j+1} = \mathcal{L}_j \cup (x, \mathcal{O}(x))$, and goes to round $j+1$. In other words, as long as there is any new query x that is ϵ-heavy to be in the closure of the queries of the view of the key-generations, **Adv** asks such a query x. If no such query exists, **Adv** breaks the loop and goes to the next step.

   (Note that the above and the following steps may require a **PSPACE**-complete oracle to be implemented efficiently.)

5. **Guessing Challenger's View:** Let $\mathcal{L}$ be the partial oracle consisting of the oracle queries (and their answers) that **Adv** learned in Steps 3 and 4 (i.e., $\mathcal{L} = \mathcal{L}_\ell$, where $\overline{Q(\mathbf{V}_\ell)}$ had no ϵ-heavy queries to be learned). Let $\mathbf{V}_{\mathbf{chal}} = (\mathbf{V} \mid \mathcal{L}, \mathsf{PK}, \{\mathsf{DK}_i\}_{i\in[k]})$, and sample $V' \xleftarrow{\$} \mathbf{V}_{\mathbf{chal}}$. Let SK' and DK'_* be in order, the "guessed" values for the secret-key and the decryption-key of f_* determined by the sampled V'. We note that by definition the other keys $\mathsf{PK}', \{\mathsf{DK}'_i\}_{i\in[k]}$ determined by V' are the same as the ones that **Adv** has received: $\mathsf{PK}, \{\mathsf{DK}_i\}_{i\in[k]}$.

[5] Step 3 can be thought of as the 0'th round.

6. **Receiving the Challenge and the Final Decryption:** **Adv** receives $C_*(= \mathbf{E}^{\mathcal{O}}(\mathsf{PK}, a_*, b))$ for a random bit $b \in \{0,1\}$. Then, **Adv** uses the oracle $\mathcal{O}'$ defined below and outputs the decrypted value $b' \leftarrow \mathbf{D}^{\mathcal{O}'}(\mathsf{PK}, \mathsf{DK}'_*, C_*)$ as his guess about the bit b.

   The Oracle $\mathcal{O}'$: At the beginning of the decryption of Step 6 the partially defined oracle $\mathcal{O}'$ is equal to $\mathcal{L} \cup Q(V')$, namely the learned queries (and their answers) together with the guessed ones specified in V'. Afterwards, if a new query x is asked:

   - if $x \in \mathcal{O}'$, return $\mathcal{O}'(x)$, otherwise
   - if $x \in \overline{\mathcal{O}'}$, then return $y = \overline{\mathcal{O}'}(x)$ and add (x, y) to $\mathcal{O}'$, and finally
   - if $x \notin \overline{\mathcal{O}'}$, ask x from $\mathcal{O}$ and add $(x, \mathcal{O}(x))$ to $\mathcal{O}'$.

4.3 Analysis of Our Attack

In this section we will show that our adversary asks only $\mathrm{poly}(\kappa)$ queries (on average), and it is able to correctly decrypt the challenge ciphertext with probability close to the completeness of the scheme τ-**TPE**.

Lemma 4.11 (Efficiency of **Adv**). *The expected number of oracle queries asked by the adversary of Construction 4.10 is at most $O(kq/\epsilon) = \mathrm{poly}(\kappa)$, where $k + 1 = \mathrm{poly}(n)$ is the number of pairs sampled in Step 1.*

Proof of Lemma 4.11. Recall that in a full execution of the algorithms $(\mathbf{G}, \mathbf{K}, \mathbf{E}, \mathbf{D})$ of τ-**TPE** the total number of queries asked is at most q. Therefore, the number of oracle queries asked by **Adv** in Steps 3 and 6 is at most $kq + q$. The only step in which the number of oracle queries is unclear is Step 4. Barak and Mahmoody [3] proved the following general lemma that can be used to argue about the expected number of queries asked in Step 4.

Lemma 4.12 ([3]). *Let A be some randomized algorithm that asks at most q oracle queries to some (potentially randomized) oracle $\mathcal{O}$ and then outputs some message K. Let B be the algorithm that receives K and keeps asking oracle queries to $\mathcal{O}$ as follows. Let $\mathcal{L}$ be the partial oracle consisting of B's knowledge about the oracle $\mathcal{O}$ at any moment. As long as there is any query $x \notin \mathcal{L}$ such that $\Pr[x \text{ asked by } A \mid K, \mathcal{L}] \geq \epsilon$, then B asks the first x satisfying this condition and adds $(x, \mathcal{O}(x))$ to $\mathcal{L}$. Then, it holds that the expected number of queries asked by B is at most q/ϵ.*

Barak and Mahmoody only (needed and) proved Lemma 4.12 for the case that $\mathcal{O}$ is a random oracle, but the very same proof holds for any oracle. In our case, the adversary **Adv** receives the partial information $\mathsf{PK}, \{\mathsf{DK}_i\}_{i \in [k]}$ from the challenger which is similar to the way B gets K from A. However, in Step 4, **Adv** keeps learning ϵ-heavy queries that are in the *closure* of the queries of the view of the challenger. To handle this issue, we can think of the algorithm A as the challenger's algorithm who is generating the keys, but A in addition also asks the dependent queries of the challenger's view from the oracle $\mathcal{O}$ so that A's view will be self-closured. This way, the total number of queries asked by A might increase by a constant factor, but it will still be at most $O(kq)$. Now we can use Lemma 4.12 to conclude that the expected number of total number of queries asked by **Adv** is at most $kq + O(kq/\epsilon) + q = O(kq/\epsilon)$. □

By stopping the adversary if his number of queries in Step 4 exceeds kq/ϵ^2, we can assume that **Adv** asks at most $O(kq/\epsilon^2) = \mathrm{poly}(\kappa)$ number of queries (in the worst case), and this will affect **Adv**'s success probability only by $O(\epsilon)$.

Now we study the success probability of (this modified) **Adv**.

Lemma 4.13 (Success of **Adv**)**.** *Let ρ be the completeness of the scheme τ-**TPE** that **Adv** is attacking, and let $t = kq/\epsilon^2 + q(k+1)$ be the upper-bound on the number of queries that are asked by* either **Adv** or *the challenger. Then,* **Adv** *will correctly decrypt the challenge ciphertext (i.e., $b' = b$) with probability at least $\rho - O(q\epsilon + t^2/s)$.*

By taking $s = \mathrm{poly}(\kappa)$ large enough one can make the success probability of **Adv** close to ρ. In the rest of this section we prove Lemma 4.13.

4.3.1 Three Experiments and Their Outputs

For the proof of Lemma 4.13 we define three experiments: Attack, Hybrid, and Ideal. The first one, Attack is the experiment in which the attack is performed, the third one, Ideal is an ideal experiment with no adversary involved, and the second one, Hybrid is a hybrid experiment that relates the probability of success in the other two experiments.

The outputs of the three experiments denoted $\mathbf{Out}_{\mathsf{att}}, \mathbf{Out}_{\mathsf{hyb}}$, and $\mathbf{Out}_{\mathsf{idl}}$ are defined with respect to the general structure of Construction 4.10 and consist of the following from the Experiment Attack, Hybrid, and Ideal, respectively: the partial oracle $\mathcal{L}$ (i.e., part of $\mathcal{O}$ that is learned in Steps 3 and 4), the sampled view V' of Step 5, and finally the view of the encryption and the decryption performed in Step 6.

Now, we proceed with the detailed description of these experiments.

- **Experiment** Attack **with output $\mathbf{Out}_{\mathsf{att}}$.** This experiment is simply the attack played by our adversary **Adv**, as defined in Construction 4.10.

- **Experiment** Hybrid **with output $\mathbf{Out}_{\mathsf{hyb}}$.** Hybrid differs from Attack as follows:

  1. The oracle $\mathcal{O}'$ is used also in the encryption of the challenge bit b (in Step 6). Note that, this way, the answers of $\mathcal{O}'$ during the decryption of the challenge might also depend on the previous answers returned by $\mathcal{O}'$ during the encryption of the challenge.
  2. The oracle $\mathcal{O}'$ never forwards its queries to $\mathcal{O}$ and instead tosses coins to answer such queries. More formally, if a new query x is asked such that $x \notin \overline{\mathcal{O}'}$, the oracle $\mathcal{O}'$ returns a random string $y \xleftarrow{\$} \{0,1\}^\lambda$, and adds $[x \mapsto y]$ to $\mathcal{O}'$. Note that $\mathcal{O}'$ does not try to respect the permutation structure of the oracle $\mathcal{O}$, but as we will see, due to the assumed condition $2^\lambda > s$, the permutation structures will be preserved with a high probability anyway.

     In other words, after learning $\mathcal{L}$ and sampling V', the oracle queries in Hybrid are answered using the lazy evaluation method as defined in Lemma 4.7 and Remark 4.8.

- **Experiment Ideal with output $\mathbf{Out}_{\mathsf{idl}}$.** This experiment differs from the previous experiments in two ways:

  1. In the step of guessing the challenger's view (Step 5), the sampled V' is chosen to be *equal* to the real view of the challenger V.

2. The oracle $\mathcal{O}'$ is simply the same as $\mathcal{O}$.

Another way to describe Ideal is to say that there is no adversary, and the challenger is generating the output $\mathbf{Out}_{\mathsf{idl}}$ by performing all the steps and simply taking $V' = V$.

At a high level, when compared to the experiments used in the proof of the main result of [11], our Attack corresponds to their Exp_0, our Hybrid corresponds to a combination of Exp_1 and Epx_2 (which, in light of our simplifications, we found it easier to study their combination directly), and finally, Ideal corresponds to their Exp_3.

Lemma 4.13 follows from the following three lemmas in a straightforward way.

Lemma 4.14 (Success in Ideal)**.** $\Pr_{\text{Ideal}}[b' = b] \geq \rho$.

Lemma 4.15 (Ideal versus Hybrid)**.** $\Delta(\mathbf{Out}_{\mathsf{idl}}, \mathbf{Out}_{\mathsf{hyb}}) \leq O(t^2/s)$.

Lemma 4.16 (Attack versus Hybrid)**.** $\Delta(\mathbf{Out}_{\mathsf{att}}, \mathbf{Out}_{\mathsf{hyb}}) \leq O(t^2/s) + O(q \cdot \epsilon)$.

Lemma 4.14 is directly implied by the ρ-completeness of the underlying τ-**TPE** scheme.

Proof of Lemma 4.15. First we note that the joint distribution of the components of $\mathbf{Out}_{\mathsf{idl}}$ and $\mathbf{Out}_{\mathsf{hyb}}$ which are sampled during Step 2 through Step 4 are identically distributed. Moreover, although V' is *not* the real view of the challenger, since it is sampled from the distribution of $\mathbf{V}$ conditioned on the previously sampled parts of the output, therefore the experiments Ideal and Hybrid will produce their (partial) outputs during Step 2 through Step 5 from the same distributions.

What remains is to compare the way the two experiments proceed in their final step of encrypting and decrypting the challenge. The only difference here is that in Hybrid (as opposed to Ideal) we are employing the lazy evaluation method to answer the oracle queries. Therefore Lemma 4.15 follows from Remark 4.8 and the assumed fact that $2^\lambda > s$. □

It only remains to prove Lemma 4.16. Without loss of generality, we make the following assumptions about the scheme τ-**TPE** and our adversary **Adv**.

Putting τ-TPE and Adv in Normal Form. In the following, we will assume w.l.o.g. that the τ-**TPE** scheme is normalized. Note that putting a scheme into normal form does not change its functionality, so the adversary can *pretend* that the scheme he is attacking is already in normal form. This way, the number of oracle queries in the protocol, in the mind of the attacker, will still be at most $O(q)$, and this will affect the bounds of Lemma 4.11 and Lemma 4.13 only by a constant factor, which because we present those bounds asymptotically, the statements will still hold without any changes whatsoever. The main point is that when the adversary considers the scheme to be in normal form, certain oracle queries that previously were *not* heavy conditioned on its own view, might become heavy now, and that makes the adversary's learning of the information about the oracle $\mathcal{O}$ relatively more effective.

We will also assume that the adversary follows the normal form restrictions *in his own learning phase* as well (i.e., Step 4). Namely, if he asks $\mathbf{k}(\mathsf{sk}, \alpha) = \mathsf{dk}$ or $\mathbf{d}(\mathsf{pk}, \mathsf{dk}, c) = m$, then he also makes the relevant additional queries to follow the rules of normal form. Note that this change in adversary's algorithm could only improve the quality of the attack while increasing the query-complexity of the attack by at most a constant factor.

Before going over the experiments Attack and Hybrid in parallel and bounding the statistical distance between them, we note that Attack and Hybrid are different only in their last step of encrypting and decrypting the challenge (i.e., Step 6). Thus, we can assume that when we get to the final step, all the previous corresponding components of the two experiments are the same, and then we study how both the experiments continue in generating their outputs. In fact, even during the execution of Step 6, we continue to assume that the two experiments have proceeded in the same way so far, and then bound the statistical distance between the answers of the next query in both the experiments *conditioned* on some "bad" events not happening. And, we also bound the probability of these bad events to be small.

We will refer to the bad events as $(\cdot)_{\mathsf{att}}, (\cdot)_{\mathsf{hyb}}$, along the comparison of the experiments Attack and Hybrid (the index of the event indicating the experiment in which it is defined). We will take the bigger bad events B_{att} and B_{hyb} to be the union of all the bad sub-events defined in Attack and Hybrid, respectively, and will bound their probabilities by bounding the probabilities of all the sub-events in each of the experiments. Then, Lemma 4.13 will follow from Lemma 2.2.

4.3.2 Defining and Bounding the Bad Events

Recall that in both experiments Attack and Hybrid, V is the real view of the key-generations, and V' is the guessed value of V sampled by **Adv** in Step 5.

Definition 4.17 (Secret Queries)**.** In the experiments Attack and Hybrid, for $U \in \{V, V'\}$, we define the secret queries of U as $\overline{Q(U)} \setminus \mathcal{L}$. Namely, these are the queries in the closure of U which the adversary didn't learn during Step 3 or the learning phase of Step 4.

Lemma 4.18. *Let* $\mathsf{HIT}_{\mathsf{att}}$ *be the event that in the experiment* Attack*, during the final encryption of Step 6, a secret query x of V' (i.e., $x \in \overline{Q(V')} \setminus \mathcal{L}$) is asked. It holds that* $\Pr[\mathsf{HIT}_{\mathsf{att}}] \le O(q\epsilon)$.

Proof. As a mental experiment, we can first finish the execution of the final encryption (which uses the oracle $\mathcal{O}$) and then go back and sample the view V' from its distribution $\mathbf{V_{chal}}$. This way, the probability of $\mathsf{HIT}_{\mathsf{att}}$ remains the same. But, since there is no ϵ-heavy query in the distribution of $\overline{Q(\mathbf{V_{chal}})} \setminus \mathcal{L}$, and since there are a total of at most q queries in the view of the final encryption, by a union bound the probability that any of them is in $\overline{Q(\mathbf{V_{chal}})} \setminus \mathcal{L}$ is at most $q\epsilon$. □

The following lemma bounds an event similar to that in Lemma 4.18, but in the experiment Hybrid. Also, here we additionally include the queries of the final decryption.

Lemma 4.19. *Let* $\mathsf{HIT}_{\mathsf{hyb}}$ *be the event that in the experiment* Hybrid*, during the final encryption* and decryption *of Step 6, a secret query x of V (i.e., $x \in \overline{Q(V)} \setminus \mathcal{L}$) is asked. It holds that* $\Pr[\mathsf{HIT}_{\mathsf{hyb}}] \le O(q\epsilon)$.

Proof. The proof is similar to that of Lemma 4.18, but since this time we are working in the experiment Hybrid, we can go ahead and finish even the final decryption, and only then come back and sample V from its distribution $\mathbf{V_{chal}}$. We could not necessarily do it similarly in Attack, because during the decryption of Attack we might forward queries to $\mathcal{O}$ due to the way $\mathcal{O}'$ is defined in Attack. By a similar union bound, the probability that any query $x \notin \mathcal{L}$ that is asked during the final encryption or decryption of Step 6, and is sampled in $\overline{Q(\mathbf{V_{chal}})}$ is at most $O(q\epsilon)$. □

Definition 4.20 (Event MAL)**.** Suppose we give the following artificial order to the queries specified in the output of our experiments:

1. Queries specified in V' that are asked during the generation of PK and DK_*.

2. Queries in $\mathcal{L}$.

3. Queries asked during the final encryption.

4. Queries asked during the final decryption.

5. Queries specified in V' that are asked during the generation of $\{\mathsf{DK}_i\}_{i\in[k]}$.

We say that the event MAL holds, if there is any triangle x_1, x_2, x_3 in the output of the experiment such that x_1 is the root, but the first appearance of x_1 according to the order above is after one of x_2, x_3. In other words, (x_1, x_2, x_3) forms a malformed triangle (see Lemma 4.9), but the order in which the queries are considered is the order above.

Lemma 4.21. *It holds that* $\Pr[\mathsf{MAL}_{\mathsf{hyb}}] = O(t^2/s)$.

Proof. In the experiment Ideal, if we ignore the learning step, one can postpone the key-generations of $\{\mathsf{DK}_i\}_{i\in[k]}$ to be done at the end of the experiment (i.e., after the final encryption and decryption are performed). This way, by using Lemma 4.9 we can conclude that $\Pr[\mathsf{MAL}_{\mathsf{idl}}] \le O(t^2/2^\lambda) = O(t^2/s)$. Unfortunately, in Hybrid we can not simply postpone the generations of $\{\mathsf{DK}_i\}_{i\in[k]}$ to the end, but by Lemma 4.15 we know that the statistical distance between the outputs of Ideal and Hybrid is $O(t^2/s)$. Therefore, we can still conclude that

$$\Pr[\mathsf{MAL}_{\mathsf{hyb}}] \le \Pr[\mathsf{MAL}_{\mathsf{idl}}] + O(t^2/s) \le O(t^2/s) + O(t^2/s) = O(t^2/s).$$

□

Definition 4.22. We say the event INT^1 (read as "intersection") holds if for some query $[\mathbf{g}(\mathsf{sk}) = \mathsf{pk}] \notin \mathcal{L}$: **(1)** $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$ is asked during the generation of PK or DK_{f_*}, **(2)** $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$ is asked during the final decryption of Step 6, and **(3)** for some $(\alpha, \mathsf{dk}) \in \{0,1\}^{2\lambda}$ the query $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$ is asked during the challenge encryption of Step 6. By INT^1_X we denote to the event INT^1 in the experiment X.

Definition 4.23. Similar to Definition 4.22, we say that the event INT^2 holds if for some query $[\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha] \notin \mathcal{L}$: **(1)** $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$ is asked during the generation of PK or DK_{f_*}, **(2)** $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$ is asked during the final decryption of Step 6, and **(3)** for some $(m, c) \in \{0,1\}^{2\lambda}$ the query $\mathbf{e}(\mathsf{pk}, \alpha, m) = c$ is asked during the challenge encryption of Step 6. By INT^2_X we denote to the event INT^2 in the experiment X.

Lemma 4.24. *For* $i = 1, 2$, *it holds that* $\Pr[\mathsf{INT}^i_{\mathsf{hyb}}] \le O(\epsilon + t^2/s)$.

Proof. We will show that $\Pr[\mathsf{INT}^2_{\mathsf{hyb}}] \le O(\epsilon + t^2/s)$, and the proof for $\Pr[\mathsf{INT}^1_{\mathsf{hyb}}] \le O(\epsilon + t^2/s)$ is identical and we describe it at the end. By Lemma 4.15 it is sufficient to show that $\Pr[\mathsf{INT}^2_{\mathsf{idl}}] \le O(\epsilon)$. As a mental experiment, we assume that the decryption-keys are generated for *every* predicate f, a random bit is encrypted under *every* attribute a, and for every $f(a) = 1$, the message encrypted under a is decrypted using the decryption-key generated for f. We refer to the latter experiment as the extended version of the experiment Ideal and the experiment Ideal can be thought of as: **(1)** executing the extended Ideal, and then (**2**) we only "look" at the encryptions and decryptions that are performed over the pairs $(a_*, f_*), (a_1, f_1), \ldots (a_k, f_k)$. We define the following sets in the extended Ideal:

- $S(a)$: The set of pairs of (pk, α) such that some query of the form $\mathbf{e}(\mathsf{pk}, \alpha, m) = c$ is asked during the encryption of the random bit under a.
- $S(f)$: The set of pairs of (pk, α) such that some query of the form $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$ is asked either during the generation of the master public-key or the generation of the decryption-key for f.
- $S(a, f)$: The set of pairs of (pk, α) such that some query of the form $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$ is asked during the decryption of the random bit encrypted under a using the decryption-key generated for f.

By the properties of the sampling algorithm Samp of Lemma 3.5 with probability $1 - \epsilon$ over the choice of $(a_*, f_*), (a_1, f_1), \dots, (a_k, f_k)$ it holds that

$$S(a_*) \cap S(f_*) \cap S(a_*, f_*) \subseteq \bigcup_{i \in [k]} S(a_i, f_i).$$

But, it is easy to see that in order for $\mathsf{INT}^2_{\mathsf{idl}}$ to hold there should be some (pk, α) that belongs to all of $S(a_*), S(f_*), S(a_*, f_*)$, and the query $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$ is *not* learned during any of the decryptions of Step 3 (because otherwise $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$ would have been in $\mathcal{L}$). But the above relation over the sets sampled by Samp implies the opposite. Therefore $\mathsf{INT}^2_{\mathsf{idl}}$ can hold only with probability ϵ.

To prove that $\Pr[\mathsf{INT}^1_{\mathsf{hyb}}] \leq O(\epsilon + t^2/s)$ we only use different definitions for the sets:

- $S(a)$: The set of pk such that some query of the form $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$ is asked during the encryption of the random bit under a.
- $S(f)$: The set of pk such that some query of the form $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$ is asked either during the generation of the public-key or the generation of the decryption-key for f.
- $S(a, f)$: The set of pk such that some query of the form $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$ is asked during the decryption under f of the random bit encrypted under a.

The rest of the proof remains the same. □

4.3.3 Comparing Experiments Attack and Hybrid

Finally, in this section we will compare the behavior of Attack and Hybrid in their final step. We will first go over the execution of the encryption of the challenge bit (and how its oracle queries are answered), and then will cover the decryption of it using oracle $\mathcal{O}'$ (which is defined *differently* in Attack and Hybrid). All we have to show is that in every oracle query, the experiments proceed $O(t/s)$-close unless one of the bad events defined in Appendix 4.3.2 happens. Then we can use Lemma 2.2 to conclude that the statistical distance between Attack and Hybrid is at most

$$\Pr[\mathsf{HIT}_{\mathsf{att}}] + \Pr[\mathsf{HIT}_{\mathsf{hyb}}] + \Pr[\mathsf{MAL}_{\mathsf{hyb}}] + \Pr[\mathsf{INT}^1_{\mathsf{hyb}}] + \Pr[\mathsf{INT}^2_{\mathsf{hyb}}] + \sum_{i \in [t]} O(t/s) = O(t^2/s) + O(q \cdot \epsilon)$$

which finishes the proof of Lemma 4.16. (The summation is done for t instead of q queries, which is the bound on the number of queries asked by τ-**TPE** algorithms, because to be able to use lazy evaluation it is important to know the number of queries asked so far, and t is the upper bound on the number of queries made by our adversary.)

1. **Encrypting the Challenge Bit.** Suppose a query x is asked during the encryption of Step 6, and Attack and Hybrid have proceeded identically so far. Here, by $\mathcal{O}$ we refer to the *partial* oracle (in both experiments) which includes only the answers to the queries that are asked so far from the oracle $\mathcal{O}$ (this also includes $Q(V)$ where V is the real view of the key-generations which is not known to **Adv**). Since $\mathcal{O}$ (resp., $\mathcal{O}'$) is used in the final encryption of Attack (resp., Hybrid), the answer to the query x in Attack (resp., Hybrid) depends on whether x is a free query with respect to $\mathcal{O}$ (resp., $\mathcal{O}'$), or not. Below we discuss the different possible cases.

   (a) If x is a free query for both of $\mathcal{O}$ and $\mathcal{O}'$: In this case, by Lemma 4.7 the answer to x in both experiments is $O(t/s)$-close to uniform over $\{0,1\}^\lambda$, and so they are $O(t/s)$-close.

   (b) If x is a free query for $\mathcal{O}'$ (i.e., $x \notin \overline{\mathcal{O}'}$, so $x \notin \mathcal{L}$), but it is *not* a free query for $\mathcal{O}$ (i.e., $x \in \overline{\mathcal{O}}$): Suppose x_1 and x_2 are the two queries in $\mathcal{O}$ that make x dependent, where x_2 is the dual query of x and x_1 is the root of the triangle.

      i. If both of x_1, x_2 are in $Q(V) \setminus \mathcal{L}$, then $x \in \overline{Q(V)} \setminus \mathcal{L}$ will be a secret query of V. Therefore, by asking x the bad event $\mathsf{HIT}_{\mathsf{hyb}}$ happens.

      ii. If none of x_1, x_2 is in $Q(V) \setminus \mathcal{L}$, then they are both in $\mathcal{O}'$, which implies that $x \in \overline{\mathcal{O}'}$ (but we assumed the other way).

      iii. If $x_1 \in Q(V) \setminus \mathcal{L}$ and $x_2 \in \mathcal{O}'$, then since x_1 is the root, due to the normal form condition of τ-**TPE**, right before asking either of x or x_2, the query x_1 is also asked to $\mathcal{O}'$. This implies the event $\mathsf{HIT}_{\mathsf{hyb}}$ happened.

      iv. If $x_2 \in Q(V) \setminus \mathcal{L}$ and $x_1 \in \mathcal{O}'$ then

         A. If x_2 is a query asked to $\mathbf{k}$ (resp., $\mathbf{d}$), then the root x_1 is a query to $\mathbf{g}$ (resp., $\mathbf{id}$), and by the normal form condition, x_1 is also present in V right before x_2. Therefore, x_1, x_2 are both in $Q(V)$, and thus $x \in \overline{Q(V)}$. This time again the query x implies the event $\mathsf{HIT}_{\mathsf{hyb}}$.

         B. If x_2 is a query asked to $\mathbf{id}$ (resp., $\mathbf{e}$), then the root x_1 is a query to $\mathbf{g}$ (resp., $\mathbf{id}$), and by the normal form condition, x_2 is asked also right after x which is of the form $\mathbf{k}$ (resp., d). The latter implies the event $\mathsf{HIT}_{\mathsf{hyb}}$.

   (c) If x is a free query for $\mathcal{O}$ (i.e., $x \notin \overline{\mathcal{O}}$, so $x \notin \mathcal{L}$), but it is *not* a free query for $\mathcal{O}'$ (i.e., $x \in \overline{\mathcal{O}}$): This case is symmetric to the previous case above and involves the bad event $\mathsf{HIT}_{\mathsf{att}}$.

   (In the remaining case in which x is not a free query for both of $\mathcal{O}$ and $\mathcal{O}'$ (i.e., $x \in \overline{\mathcal{O}}, x \in \overline{\mathcal{O}'}$), the answer to x comes from identical distributions in both experiments.)

2. **Decrypting the Challenge Bit.** Now suppose a query x is asked during the decryption of Step 6, and Attack and Hybrid have proceeded identically so far (including the encryption of the challenge). A key point is that during the decryption, the oracle $\mathcal{O}'$ might have different definitions in Attack and Hybrid (even if they proceed identically) because $\mathcal{O}'$ in Attack does not have access to the view of the final encryption. Therefore, we use $\mathcal{O}'_{\mathsf{att}}$ and $\mathcal{O}'_{\mathsf{hyb}}$ to distinguish them. We also define the partial oracle $\mathcal{O}''$ defined in both experiments as follows. At any time, $\mathcal{O}''$ consists of $Q(V) \cup \mathcal{L}$ as well as the queries in the view of the final encryption and the decryption so far. Note that, in our parallel comparison, we also assume that the oracle $\mathcal{O}''$ is identical across Attack and Hybrid so far. Below we discuss the different possible cases.

(a) If $x \in \overline{\mathcal{O}'_{\mathsf{att}}}$, since $\mathcal{O}'_{\mathsf{att}} \subseteq \mathcal{O}'_{\mathsf{hyb}}$, therefore it holds that $x \in \overline{\mathcal{O}'_{\mathsf{hyb}}}$ as well and the same answer is used in both experiments.

(b) If $x \notin \mathcal{O}'_{\mathsf{att}}$:

   i. If $x \in \overline{\mathcal{O}'_{\mathsf{hyb}}}$: due to the details of this case, we continue this case below.

   ii. If $x \notin \overline{\mathcal{O}'_{\mathsf{hyb}}}$:

      A. If $x \notin \overline{\mathcal{O}''}$: In Hybrid, x is a free query because of $x \notin \overline{\mathcal{O}'_{\mathsf{hyb}}}$. In Attack, all the queries asked to $\mathcal{O}$ so far are included in $\mathcal{O}''$ and so $x \notin \overline{\mathcal{O}''}$ implies that x is a free query in Attack as well. Therefore x is a free query in both experiments and its answer is chosen from (t/s)-close distributions.

      B. If $x \in \overline{\mathcal{O}''}$: The case study of this case is identical to Case 1b above.

Continuing Case 2(b)i:

Let x_1 and x_2 be the other two queries that make a triangle with x in $\mathcal{O}'$, and let x_1 be the root of the triangle and x_2 the dual query of x. Since this triangle does not exist in $\mathcal{O}'_{\mathsf{att}}$, at least one of x_1 and x_2 should be asked during the final encryption.

- If x_1 is asked during the final encryption:
  - If $x_2 \in Q(V') \setminus \mathcal{L}$: In this case event $\mathsf{MAL}_{\mathsf{hyb}}$ has happened.
  - If $x_2 \notin Q(V') \setminus \mathcal{L}$: then in Attack both of x_1, x_2 are asked to $\mathcal{O}$. Since in Attack, x is also forwarded to $\mathcal{O}$, it follows that the same answer determined by the answers of x_1, x_2 is used for x in both experiments Attack and Hybrid.
- If x_2 is asked during the final encryption:
  - If $x_1 \notin Q(V') \setminus \mathcal{L}$: In this case again in Attack all of x_1, x_2 and x are asked to $\mathcal{O}$ and so the same answer determined by the answers of x_1, x_2 is used for x in both experiments Attack and Hybrid.
  - If $x_1 \in Q(V') \setminus \mathcal{L}$: Let V'_* be the union of the views of the challenger during the generation of PK and DK_* as specified in V', and let V'_j for $j \in [k]$ be the challenger's view when generating DK_j as specified in V'.
    - ∗ If $x_1 \notin Q(V'_*)$: Then x_1 should belong to $Q(V'_j)$ for some $j \in [k]$, and so the event $\mathsf{MAL}_{\mathsf{hyb}}$ has happened.
    - ∗ If $x_1 \in Q(V'_*)$: Then x_2 can not be a $\mathbf{k}$ or $\mathbf{d}$ query, because these queries would make x_1 to be asked during the final encryption and that implies the event $\mathsf{HIT}_{\mathsf{att}}$ has happened.
      - · If x_1 is of the form $\mathbf{g}(\mathsf{sk}) = \mathsf{pk}$, then x_2 should be of the form $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$, and x of the form $\mathbf{k}(\mathsf{sk}, \alpha) = \mathsf{dk}$. By the normal form condition x_1 is asked right before x, at which point (since $x_1 \in Q(V'_*)$) the event $\mathsf{INT}^1_{\mathsf{hyb}}$ has happened.
      - · If x_1 is of the form $\mathbf{id}(\mathsf{pk}, \mathsf{dk}) = \alpha$, then x_2 should be of the form $\mathbf{e}(\mathsf{pk}, \alpha, m) = c$, and x of the form $\mathbf{d}(\mathsf{pk}, \mathsf{dk}, c) = m$. By the normal form condition x_1 is asked right before x, at which point (since $x_1 \in Q(V'_*)$) the event $\mathsf{INT}^2_{\mathsf{hyb}}$ has happened.

Acknowledgement. We thank Brent Waters for suggesting to us the problem of separating predicate encryption from IBE and for useful collaborations in the initial stages of this project.

References

[1] Nuttapong Attrapadung and Benoît Libert. Functional Encryption for Inner Product: Achieving Constant-Size Ciphertexts with Adaptive Security or Support for Negation. In *Public Key Cryptography*, pages 384–402, 2010. 1

[2] László Babai, Peter Frankl, and Janos Simon. Complexity classes in communication complexity theory (preliminary version). In *FOCS*, pages 337–347, 1986. 13

[3] Boaz Barak and Mohammad Mahmoody. Lower bounds on signatures from symmetric primitives. In *FOCS: IEEE Symposium on Foundations of Computer Science (FOCS)*, 2007. 4, 24

[4] Boaz Barak and Mohammad Mahmoody. Merkle puzzles are optimal - an $O(n^2)$-query attack on any key exchange from a random oracle. In Shai Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 374–390. Springer, 2009. 4

[5] Mihir Bellare and Phillip Rogaway. *Introduction to Modern Cryptography.* Lecture Notes. `http://cseweb.ucsd.edu/users/mihir/cse207`. 8

[6] Mihir Bellare and Phillip Rogaway. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993. 17

[7] Alexandra Boldyreva, Vipul Goyal, and Virendra Kumar. Identity-based encryption with efficient revocation. In *ACM Conference on Computer and Communications Security*, pages 417–426, 2008. 1

[8] Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In *EUROCRYPT*, pages 223–238, 2004. 2

[9] Dan Boneh and Xavier Boyen. Secure Identity Based Encryption Without Random Oracles. In *CRYPTO*, pages 443–459, 2004. 2

[10] Dan Boneh and Matthew K. Franklin. Identity-Based Encryption from the Weil Pairing. *SIAM J. Comput.*, 32(3):586–615, 2003. 1, 2

[11] Dan Boneh, Periklis A. Papakonstantinou, Charles Rackoff, Yevgeniy Vahlis, and Brent Waters. On the Impossibility of Basing Identity Based Encryption on Trapdoor Permutations. In *FOCS*, pages 283–292, 2008. 2, 3, 4, 7, 16, 22, 26

[12] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In *TCC*, pages 253–273, 2011. 1, 6

[13] Dan Boneh and Brent Waters. Conjunctive, Subset, and Range Queries on Encrypted Data. In *TCC*, pages 535–554, 2007. 1

[14] Clifford Cocks. An Identity Based Encryption Scheme Based on Quadratic Residues. In *IMA Int. Conf.*, pages 360–363, 2001. 1, 2

[15] Eiichiro Fujisaki and Tatsuaki Okamoto. How to Enhance the Security of Public-Key Encryption at Minimum Cost. In *Public Key Cryptography*, pages 53–68, 1999. 17

[16] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure Integration of Asymmetric and Symmetric Encryption Schemes. In *CRYPTO*, pages 537–554, 1999. 17

[17] Craig Gentry. Practical identity-based encryption without random oracles. In *EUROCRYPT*, pages 445–464, 2006. 2

[18] Vipul Goyal. Reducing Trust in the PKG in Identity Based Cryptosystems. In *CRYPTO*, pages 430–447, 2007. 1

[19] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM Conference on Computer and Communications Security*, pages 89–98, 2006. 1

[20] Russell Impagliazzo and Steven Rudich. Limits on the Provable Consequences of One-Way Permutations. In *STOC*, pages 44–61, 1989. 1, 17

[21] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In *EUROCRYPT*, pages 146–162, 2008. 1

[22] Jonathan Katz and Arkady Yerukhimovich. On Black-Box Constructions of Predicate Encryption from Trapdoor Permutations. In *ASIACRYPT*, pages 197–213, 2009. 9

[23] Hartmut Klauck. Rectangle Size Bounds and Threshold Covers in Communication Complexity. In *IEEE Conference on Computational Complexity*, pages 118–134, 2003. 14

[24] Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997. 13

[25] Troy Lee and Adi Shraibman. Lower Bounds in Communication Complexity. *Foundations and Trends in Theoretical Computer Science*, 3(4):263–398, 2009. 13

[26] Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption. In *EUROCRYPT*, pages 62–91, 2010. 1

[27] Satyanarayana V. Lokam. Spectral Methods for Matrix Rigidity with Applications to Size-Depth Trade-offs and Communication Complexity. *J. Comput. Syst. Sci.*, 63(3):449–473, 2001. 15

[28] Satyanarayana V. Lokam. Complexity Lower Bounds using Linear Algebra. *Foundations and Trends in Theoretical Computer Science*, 4(1-2):1–155, 2009. 13

[29] Mohammad Mahmoody-Ghidary and Avi Wigderson. Black Boxes, Incorporated., 2009. `http://www.cs.cornell.edu/~mohammad/files/papers/BlackBoxes.pdf`. 1

[30] Tatsuaki Okamoto and Katsuyuki Takashima. Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption. In *CRYPTO*, pages 191–208, 2010. 1

[31] Adam ONeill. Denitional issues in functional encryption, 2010. http://www.cs.cornell.edu/ mohammad/files/papers/BlackBoxes.pdf. 1

[32] Alexander Razborov. On Rigid Matrices (in Russian), 1989. http://people.cs.uchicago.edu/~razborov/rigid.pdf. 15

[33] Omer Reingold, Luca Trevisan, and Salil P. Vadhan. Notions of reducibility between cryptographic primitives. In *Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004*, volume 2951 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2004. 1

[34] Amit Sahai and Brent Waters. Fuzzy Identity-Based Encryption. In *EUROCRYPT*, pages 457–473, 2005. 1, 2, 7

[35] Adi Shamir. Identity-Based Cryptosystems and Signature Schemes. In *CRYPTO*, pages 47–53, 1984. 1, 7

[36] Brent Waters. Efficient identity-based encryption without random oracles. In *EUROCRYPT*, pages 114–127, 2005. 2